C= újság

Az Országos Commodore Egyesület lapja

1992/10

Fizikafeladat — számítógéppel



Ismét a

BŐVÍTŐK

Cserélhető lemezes winchester!



A SyQuest Technology hivatalos magyarországi disztribútora:

NOVOTRADE SZERVIZ Kft.

Cím: 1053 Budapest, Henszlmann I. u. 9.
Telefon: 117-4144 Telefax: 117-9692

SQ 555 (meghajtó)	**39 900 Ft**
SQ 400 (44 MB/lemez)	**8 400 Ft**
SQ 5110 (meghajtó)	**61 400 Ft**
SQ 800 (88 MB/lemez)	**11 900 Ft**

Áraink végfelhasználói árak és ÁFA-t nem tartalmaznak.

Meghatalmazott dealereink:

Microteam Kft., 1145 Budapest, Róna u. 127.
Tel./fax: 184-1226

Professzionál Kft., 1033 Budapest, Kaszásdůlő u. 5.
Tel.: 167-0024 fax: 167-0289

Professzionál Kft., Miskolci Kirendeltség,
3525 Miskolc, Szabó L. u. 37.
Tel.: 46/56-079

Professzionál Kft., Békéscsabai Kirendeltség,
5600 Békéscsaba, Andrássy u. 75.
Tel.: 66/28-584

Korall Kft., 2800 Tatabánya, Március 15. út 3.
Tel.: 34/11-714

3S Computer Kisszövetkezet, 6723 Szeged, Kemes u. 6.
Tel.: 62/26-277, Fax: 62/26-347

Volánelektronika Vercomp Kft., 9024 Győr, Dr. Petz Lajos u. 7.
Tel./fax: 96/12-520

Novotrade Miskolc Kft., 3530 Miskolc, Vörösmarty u. 51.
Tel./fax: 46/49-489

Elektrosoft Kft., 5000 Szolnok, József A. u. 6—8.
Tel.: 56/42-880, Fax: 56/44-222

Számadó Kft., 6000 Kecskemét, Dózsa Gy. u. 29.
Tel.: 76/21-455, Fax: 76/21-462

BX-Next Kft., 3434 Mályi, Bercsényi út 50.
Tel.: 46/91-117

Onyx Szoftver Kft., 1118 Budapest, Mányoki u. 14/B.
Tel.:165-3325

Novotrade PC Kft., 1136 Budapest, Sallai u. 25.
Tel.: 149-0798 Fax: 131-0734

Almárium Kft., 1137 Budapest, Pozsonyi u. 21—23.
Tel.: 111-2830 Fax: 112-3647

Microchip Kft., 8000 Székesfehérvár, Élmunkás u. 47.
Tel.: 22/25-514

Navigátor Kft., 4400 Nyíregyháza, Kórház út 26/B.
Tel./fax: 42/41-972

Alfadat Kft., 2803 Tatabánya, Tóth-Bucsoki út 12.
Tel.: 34/10-234, 10-405, Fax: 34/10-729

Digitech Kft., 7101 Szekszárd, Rákóczi u. 6.
Tel.: 74/16-874

Interface Kft., 1116 Budapest, Hunyadi J. út 162.
Tel.: 166-5322/58, 55, Fax: 226-3793

Netrend Rt., 1089 Budapest, Elnök u. 1.
Tel.: 113-8217, Fax: 113-9537

Omnis Kft., 2840 Oroszlány, Münnich F. u. 23.
Tel.: 34/60-832

2R Periféria Kft., 1071 Budapest, Peterdy u. 30.
Tel.: 122-3034 Fax: 142-3308

ProComp Kft., 8900 Zalaegerszeg, Bíró M. út 8. Pf.: 275.
Tel.: 92/11-373

A magyar olimpiai csapat arany fokozatú támogatója

SZERENCSEJÁTÉK RT

MIT, HOGYAN, HOL, MIKOR?

EGYESÜLETI ÜGYEK: Egyesületünknek tagja lehet mindenki, aki a tagsági díjat befizeti. A tagdíjat személyesen az egyesület irodájában (1025 Budapest, Vöröstorony utca 29. Telefon: 1-76-22-57), vagy átutalással az MNB 217-98 292, OTP 565-3610-8 számlára lehet befizetni. Megrendelés esetén számlát küldünk.

Pötyögőszolgálatunk valamint a szervizkedvezmény és az apróhirdetés lehetősége tagjaink rendelkezésére áll.

A DEÁKPÁHOLY tagjai minden hónapban megkapják a C-újságot, a tagsági díj egy egy évre 777 forint.

A PLUSZPÁHOLY tagjai minden hónapban megkapják a C-újságot, és kapnak havonta 3 db vásárlási utalványt. A tagsági díj egy évre 1888 Ft.

A SZUPERPÁHOLY tagjai havonta 15 példányt kapnak a C-újságból, és ezzel havonta 15x3 db vásárlási utalványt is. Az éves tagsági díj 20 900 Ft.

ÜGYFÉLFOGADÁS: Minden kedden és csütörtökön 12—16 óra között várjuk tagjainkat és az érdeklődőket.

PÖTYÖGŐSZOLGÁLAT: Az újságban megjelenő programokat másolja a megrendelők részére. Megrendelhető személyesen az egyesület irodájában vagy postai utánvéttel. Postacím: 1388 Budapest 62., Postafiók: 86.

APRÓHIRDETÉS: Az egyesületi tagoknak ingyen áll rendelkezésére. Nem tagoknak a hirdetés ára 100 forint. A hirdetés módja: az újságban megjelenő nyomtatvány kitöltésével.

A C-ÚJSÁG RÉGEBBI SZÁMAI megvásárolhatók az egyesület irodájában, vagy megrendelhetők utánvéttel.

Kedvezményes ár! Tagoknak olcsóbb!

Az újságban eddig megjelent programok gépenként összegyűjtve megrendelhetők. VC 20, C16, PLUS/4, C128, C64. További felvilágosítást is adunk a 1-76-22-57-es telefonszámon vagy levélben!

Vidéki pluszpáholy-tagjaink háromhavi tikett összegyűjtésekor igénybe vehetik a NOVOTRADE 2C Áruház csomagküldő szolgálatát.

VIDÉKEN TOVÁBBI INFORMÁCIÓK KAPHATÓK:
Baja, AXIS Kft.,
Győri Bartók Béla Művelődési Ház,
Jászberényi Városi Könyvtár,
Kecskemét, SZIGMA—BIT,
Pécsi Apáczai Csere János Gimnázium,
Zalaegerszegi Ságvári Endre Gimnázium.

Az Országos Commodore Egyesület módszertani kiadványa

Egyesületi iroda és szerkesztőség:
1025 Budapest, Vöröstorony utca 29. Telefon: 1-76-22-57
Felelős kiadó: Horváth Judit, az egyesület elnöke
Főszerkesztő: Rados Péter, az OCE főtitkára
Felelős szerkesztő: dr. Horváth András
Művészeti szerkesztő: Bausz Sándor
Levélcím: Commodore Újság, 1388 Budapest, 62. Pf.: 86.
Index: ISSN 0237-756 X
Terjeszti a Magyar Posta
Megvásárolható a hírlapárusoknál
92.0305 MSZH Nyomda és Kiadó Kft., Budapest
Felelős vezető: Nagy László

Pályázat

Kedves Egyesületi tagok, tisztelt olvasóink!

Egyesületünk ismét pályázat kiírását határozta el. Azt szeretnénk, ha a pályaművek ezúttal a lap minden olvasója számára érdekesek és hasznosak lennének és nemcsak a bírálók örömét (vagy bosszúságát?) szolgálnák. Ezért olyan pályázatokat várunk, melyek lapunkban teljes terjedelemben leközölhetők, az olvasó által jól áttekinthetők, vagyis — Commodore-ról lévén szó — BASIC programokat. Egy nyomtatásban közölt program akkor igazán értékes, ha nemcsak használatát, hanem működését is leírja a szerző.

A pályázat témáját tekintve először arra gondoltunk, kiválasztunk egy konkrét stratégiai játékot és a beérkezett pályaművek között bajnokságot rendezünk. Ez azonban éppen a működési leírások értékének figyelembe vételét nehezítené meg, ezen kívül a sok azonos témájú program végigolvasása talán unalmas is lenne. Ezek alapján született az alábbi kiírás!

Az Országos Commodore Egyesület pályázatot hirdet. A pályázaton bárki részt vehet. A pályázat tárgya: C64-en vagy C+4-en futó, saját készítésű, stratégiai játékokat játszó program. A program teljes egészében a gép saját, bővítés nélküli BASIC nyelvén készüljön, nem tartalmazhat, nem állíthat elő és nem hívhat meg gépi kódú programrészeket. A programot kérjük lehetőleg lemezen (esetleg kazettán) beküldeni. A pályázatnak tartalmaznia kell a játék szabályait, a szerző által alkalmazott stratégiát és egy részletes programleírást, mely bemutatja, hogy a program egyes részei hogyan valósítják meg a választott stratégiát.

A pályázat díjazása:

1—2 díj: C64-es számítógép, illetve nyomtató.
3—5. díj: C64-es bővítők a Hobbi Elektronika választékából, lapelőfizetés, lemezek, programok.
A 6—10. helyezettek jutalmat kapnak!
A pályázat beküldési határideje: 1992. október 15.

Javasoljuk pályázóinknak, lehetőleg ne túl bonyolult (mint sakk vagy go) játékot válasszanak, az sem baj, ha nem túl közismert. A témaválasztáshoz segítséget nyújthat Nowak :50 táblás játék című könyve (Gondolat 1982). Hódi Gyula jóvoltából ennek alapján közreadunk egy játékszabály mintát. Hangsúlyozzuk, ez csak minta, tetszőleges más játék is választható.

OCE

Minta

Szidzsa

Eszközök egy 7×7 mezős tábla, a középső mező megjelölve, és 24–24 bábu (világos és sötét).

Alapállásban a tábla üres.

A játék célja az ellenfél bábuinak kiütése.

Első rész: a játékot világos kezdi. A játékosok felváltva egy-egy bábut helyeznek a tábla egy szabad mezőjére. A középső mezőre bábut helyezni tilos. A játék első részében ütni nem lehet.

Második rész: az első lépést sötét teheti meg.

Menetmód: egy lépés egy szabad szomszédos vagy sarkosan érintkező mezőre (8 irány). Az első lépés a tábla középső mezőjére történik. Ha egy játékosnak nincs lépési lehetősége, akkor ellenfele következik.

Kiütés: ellenséges bábu úgy vehető le, hogy két saját bábuval közrefogjuk, azaz lépésünkkel olyan helyzet keletkezik, ahol egy vonalban saját-ellenséges-saját bábu van, egymással szomszédos (közös oldalú) mezőkön. Egy lépéssel csak egy bábu üthető ki. Ütést eredményező lépés után a játékos újabb lépést tehet.

Nem számít ütésnek, ha az ellenséges bábu lép olyan mezőre, amelyet két saját bábu vesz közre. Ilyenkor az ellenséges bábu nem vehető le.

Nem üthető ki a középső mezőn álló bábu.

A játéknak vége: ha az egyik játékosnak csak egy bábuja maradt. Ellenfele győzött.

Tisztelt Szerkesztőség

A C64-en létező szöveg- és kiadványszerkesztők közül szerintem a Printfox a legjobb. Ehhez a programcsomaghoz tartozik a Character Fox, amellyel a karakterkészletek szerkeszthetők. Ennek használata közben felmerült a gond, hogy ékezeteket akartam írni a betűkre, de nem volt fölöttük elég hely. Eddig még nem láttam olyan programot, amely segíthetett volna megnövelnem a karakterek képe számára eredetileg megállapított területet. Az itt látható program lényegében erre szolgál.

A karakterkészlet négy paramétere állítható. Első a karakterek száma, második a karaktermező pontoszlopainak száma (hossz, maximal length). Következik a sorok száma, a karakterhely teljes magassága (total height), végül az ún. belső magasság (internal height). Ez utóbbiban nincs benne az alapvonal alatti sorok száma, ahova például az 'y' szára lenyúlik.

A program alkalmazható a készlet átnevezésére is, amely a szokásos módon nem oldható meg.

Tesztelés közben rájöttem, hogy a Printfox programok csak olyan készletet tudnak hibátlanul betölteni, amelynek a teljes (tömörítetlen) hossza legfeljebb 8 kilobyte, ellentétben a 14 kB elvi maximummal. A túlcsorduló karakterek nem használhatók. A program ezért figyelmeztet a túlcsordulásra, de hibátlanul kimenti a teljes karakterkészletet.

A BASIC forráskódot elemezve megismerhető a karakterkészletek kódolási és tömörítési rendszere. Gyakori használathoz a program kissé lassú, de a Pötyögőszolgálat számára egy gyorsított és lefordított változatot is mellékeltem.

Üdvözlettel: Dave (HBC)

```
100 REM *** PRINTFOX ZS RESIZER *
110 REM (C) DAVE - 9202
120 POKE 788,52
130 POKE 53280,0: POKE 53281,0
140 PRINT "  PRINTFOX ZS RESIZER V1.1  BY DAVE"
150 DIM C$(114,3),L(114)
160 Z$=CHR$(0): ZX$="": FOR I=1 TO 42: ZX$=ZX$+Z$: NEXT I: T=155
170 DEF FNL(X)=8*(INT((X-1)/8)+1)
180 REM ------ LOAD
190 PRINT "ENTER THE NUMBER OF CHARSET TO RESIZE!"
200 PRINT ,"(1-255, 0 TO QUIT):";: W=0: GOSUB 1330: IF W=0 THEN 1100
210 IF W>255 THEN PRINT ,"INCORRECT NUMBER!": GOTO 190
220 ZS$="ZS"+MID$(STR$(W),2): ZS=W: PRINT "CHECKING "ZS$" ON DISK"
230 OPEN 15,8,15,"I0": INPUT#15,E,E$: IF E THEN 1120
240 OPEN 2,8,2,ZS$+",P,R": INPUT#15,E,E$: IF E THEN 1120
250 REM ---
260 PRINT "LOADING...": POKE 144,0
270 GOSUB 1220: IF V<>90 THEN 1110
280 GOSUB 1220: IF V<>ZS THEN 1110
290 GOSUB 1220: L=V
300 GOSUB 1220: H=V
310 GOSUB 1220: N=V-32: IF N<1 OR N>114 THEN 1110
320 ML=0: FOR I=1 TO 114: GOSUB 1220: IF I>N THEN V=0
330 IF V>ML THEN ML=V
340 L(I)=V: NEXT I: PRINT
350 GOSUB 1220: B=V: GOSUB 1220: C=0
360 REM ---
370 M0=H+1: M1=0: FOR I=1 TO N: PRINT ""N-I+1" "
380 FOR J=1 TO H: FOR K=1 TO L: GOSUB 1180
390 C$(I,K)=C$(I,K)+CHR$(V): IF V THEN IF M1<J THEN M1=J
400 IF V THEN IF M0>J THEN M0=J
410 NEXT K: NEXT J: NEXT I: PRINT "  ": CLOSE 2: CLOSE 15
420 IF ST<64 THEN PRINT " EXTRA BYTES IGNORED": FOR I=1 TO 2000: NEXT I
430 REM ------ REPORT
440 PRINT "          CHARSET REPORT "ZS$
450 PRINT "     NUMBER OF CHARACTERS:"N
460 PRINT "          MAXIMAL LENGTH:"L*8"COLUMNS"
470 PRINT "        EFFECTIVE LENGTH:"FNL(ML)"("MID$(STR$(ML),2)")"
480 PRINT "            TOTAL HEIGHT:"H"ROWS"
490 PRINT "         (INTERNAL HEIGHT:"B")"
500 PRINT "HIGHEST SIGNIFICANT ROW:"M0
510 PRINT " LOWEST SIGNIFICANT ROW:"M1
520 PRINT "        EFFECTIVE HEIGHT:"(M0-M1-1)*(M1>=M0)
530 PRINT "          MEMORY REQUEST:"INT(N*L*H/8192*100+.9999)"%"
540 REM ---
550 PRINT "          ENTER NEW PARAMETERS!"
560 PRINT "NUMBER OF CHARACTERS (1-114):";: W=N: GOSUB 1330
570 IF W<1 OR W>114 THEN PRINT "": GOTO 560
580 NN=W
```

```
590 PRINT ,"■MAXIMAL LENGTH (1-24):";: W=ML: GOSUB 1330
600 IF W<1 OR W>24 THEN PRINT "■": GOTO 590
610 NL=W
620 PRINT ,"■TOTAL HEIGHT (5-42):";: W=H: GOSUB 1330
630 IF W<5 OR W>42 THEN PRINT "■": GOTO 620
640 NH=W
650 PRINT "■       INTERNAL HEIGHT (4-42):";: W=(H-B-NH)*(B+NH>H): GOSUB 1330
660 IF W<4 OR W>42 THEN PRINT "■": GOTO 650
670 NB=W
680 REM ---
690 PRINT "■","■NEW PARAMETERS:■"
700 PRINT "■     NUMBER OF CHARACTERS:■"NN
710 PRINT ,"■   TOTAL LENGTH:■"NL"■(■"MID$(STR$(FNL(NL)),2)"■)"
720 PRINT ,"■   TOTAL HEIGHT:■"NH
730 PRINT ,"■INTERNAL HEIGHT:■"NB
740 V=INT(NN*FNL(NL)/8*NH/8192*100+.9999)
750 V$="■": IF V>100 THEN V$="■"
760 PRINT ,"■ MEMORY REQUEST:"V$;V"■%"
770 IF V>100 THEN PRINT ,,"■OVERFLOW WARNING!"
780 PRINT ,"■ARE THESE CORRECT? (■Y■/■N■)"
790 GOSUB 1450: IF V=78 THEN 440
800 IF V<>89 THEN 790
810 NL=FNL(NL)/8: D0=B-NB+1: D1=D0+NH-1
820 REM ------ SAVE
830 CLOSE 2: CLOSE 15: PRINT "■ ENTER THE NUMBER OF THE NEW CHARSET!■"
840 PRINT ,"■(1-255, 0 TO QUIT):■";: W=ZS: GOSUB 1330: IF W=0 THEN 1100
850 IF W>255 THEN PRINT ,"■INCORRECT NUMBER!■": GOTO 830
860 ZS=W: ZS$="ZS"+MID$(STR$(W),2): PRINT "■CHECKING ■"ZS$"■ ON DISK■"
870 OPEN 15,8,15,"I0": INPUT#15,E,E$: IF E=0 THEN 890
880 GOSUB 1160: GOTO 830
890 OPEN 2,8,2,ZS$+",P,R": INPUT#15,E,E$: CLOSE 2: IF E=62 THEN 960
900 IF E THEN GOSUB 1160: GOTO 830
910 PRINT "■ FILE EXISTS. OVERWRITE? (■Y■/■N■)■"
920 GOSUB 1450: IF V=78 THEN 830
930 IF V<>89 THEN 920
940 PRINT#15,"S0:"+ZS$: INPUT#15,E
950 REM ---
960 PRINT "■SAVING...■": OPEN 2,8,2,ZS$+",P,W": INPUT#15,E,E$: IF E THEN 880
970 PRINT#2,"Z"CHR$(ZS)CHR$(NL)CHR$(NH)CHR$(NN+32);
980 L=NL*8: FOR I=1 TO 114: V=L(I): IF I>NN THEN V=0
990 IF V>L THEN V=L
1000 PRINT#2,CHR$(V);: NEXT I: PRINT#2,CHR$(NB)Z$;: PRINT
1010 REM ---
1020 F=2: FOR I=1 TO NN: PRINT "■"NN-I+1"■ "
1030 FOR J=D0 TO D1: IF J>0 THEN 1050
1040 FOR K=1 TO NL: V=0: GOSUB 1250: NEXT K: GOTO 1060
1050 FOR K=1 TO NL: V=ASC(MID$(C$(I,K)+ZX$,J)): ON F GOSUB 1270,1250: NEXT K
1060 NEXT J: NEXT I
1070 GOSUB 1280: PRINT "■ ■": CLOSE 2: INPUT#15,E,E$: GOSUB 1160
1080 REM ------ EXIT
1090 PRINT "■ PROGRAM TERMINATED.■": GOTO 1140
1100 PRINT "■ PROGRAM ABORTED BY USER'S REQUEST.■": GOTO 1140
1110 PRINT " ■FILE DATA ERROR": GOTO 1130
1120 GOSUB 1160
1130 PRINT "■ PROGRAM ABORTED.■"
1140 CLOSE 2: CLOSE 15: POKE 788,49: END
1150 REM ====== ERROR
1160 PRINT "■DISK REPORT:■"E"- "E$: RETURN
1170 REM ====== READ (O:V;C)
1180 IF C THEN C=C-1: RETURN
1190 GOSUB 1220: IF V<>T THEN RETURN
1200 GOSUB 1220: C=V: GOSUB 1220: C=C+V*256-1: GOSUB 1220: RETURN
1210 REM ====== GET# (O:V)
1220 IF ST THEN 1110
1230 GET #2,V$: V=ASC(V$+Z$): RETURN
1240 REM ====== WRITE (I:V;C,F)
1250 F=1: GOTO 1310
1260 REM ===
```

```
1270 IF V=P THEN C=C+1: RETURN
1280 IF P=T THEN 1300
1290 IF C<5 THEN FOR E=1 TO C: PRINT#2,CHR$(P);: NEXT E: GOTO 1310
1300 PRINT#2,CHR$(T)CHR$(C AND 255)CHR$(C/256)CHR$(P);
1310 C=1: P=V: RETURN
1320 REM ====== NUMEDIT (I/O:W)
1330 W$="": A=PEEK(211): PRINT W"▊  ";: POKE 211,A+1
1340 GOSUB 1440: IF V=13 THEN 1410
1350 IF V=20 AND W$>"" THEN W$=LEFT$(W$,LEN(W$)-1): GOTO 1390
1360 IF V<48 OR V>57 THEN 1340
1370 IF LEN(W$)=3 THEN 1340
1380 W$=W$+V$
1390 PRINT V$"   ▊▊▊▊";: GOTO 1340
1400 REM ---
1410 IF W$>"" THEN W=VAL(W$)
1420 POKE 211,A: PRINT W"▊ ": RETURN
1430 REM ====== GET (O:V,V$)
1440 POKE 204,0
1450 WAIT 198,15: GET V$: V=ASC(V$): POKE 204,1: RETURN
```

Tisztelt Szerkesztőség

Elég gyakran találkozhatunk a nagyobb programokban azzal, hogy a speciális feladatokat végrehajtó, gyakran használt basic rutinokat a program végén helyezik el ötszámjegyű (10000 felett) sorszámmal. Bizonyára mindenkinek van ilyen készített (kedvenc) vagy szerzett programja. A BASIC programozás magasiskolája című kiadvány is ilyen nagysorszámú rutinokat ajánl!

Töltse be az olvasó az ilyen programot, és tanulmányozza a memóriát az alábbi sorral parancs módban:

KC=2048:FORI=KCTOKC+800:POKE646,14:PRINTCHR$(PEEK(I));:NEXT

KC értéke: 2048 és 40959 (–800) között.

Sajnos a vezérlő CHR$(X) kódok PRINT-el végrehajtódnak, nem íródnak ki a képernyőre, ezért ezek az értékek hiányoznak. POKE646,14-el a színek az olvashatóság kedvéért visszaállítva.

A képernyőn tanulmányozható memóriatartalom elegendő a mondanivalóm alátámasztására. Látható, hogy a BASIC sorszámok alsó- és felsőbit formában szerepelnek a sor elején, de az ugrási sorértékek eredetiben. Ez azt jelenti, hogy öt számjegy helyigénye öt bit. A következtetés: rövidebb lehet a program, ha a rutinok 1–2 számjegyből álló sorokban találhatók.

Célravezető módszernek látszik a programok rövidítésére: a sokszor használt rutinok részére ugrótábla elhelyezése a program elején.

```
0 REM BASIC UGROTABLA
1 GOTO100:REM PROGRAMKEZDES
2 GOTO10000:REM KIIRAS
3 GOTO11000:REM TORLES
4 GOTO12000:REM BEVITEL
5 GOTO13000:REM KERESES
6 REM STB.
. . . . . . . . . . . .
100 CLR:PRINTCHR$(147)
. . . . . . . . . . . .
212 GOSUB3:GOSUB2:REM TORLES KIIRAS
10000 REM KIIRAS
10001 REM STB.
10050 RETURN
11000 REM TORLES
11001 REM STB.
11045 RETURN
```

Az ugrótábla sora GOSUB-al hívható, végrehajtás (RETURN) után a GOSUB mögé térünk vissza.

Szerkesztés közben is segítséget nyújt ez a megoldás, mert áttekinthető; csak az ugrótáblát kell kinyomtatni, vagy leírni. Nem kell keresni a rutinokat, mert azok jegyzéke maga az ugrótábla. A program futását elméletileg lassítja ez a megoldás, hiszen egy ugrást beiktat, ám ha a program rövidebb az gyorsulást eredményez!

Íme, ismét egy ötlet, amin érdemes elgondolkodni, és bevinni a BASIC programozás kelléktárába!

Egy nyomtatási műveletet talán hasznos lenne közismertté tenni: BASIC programok listázása leporellón CITIZEN 120D-el.

1. Töltse be a listázandó programot;
2. Helyezze be a nem leszabott leporellót úgy, hogy a nyomófej tetején legyen a papírszél;
3. Nyomja le az LF és hozzá az ON LINE gombokat (szépírásmód választás);
4. Írja a képernyőre parancs módban az alábbiakat:

Megjegyzés: ⟨R⟩=return leütése

```
OPEN 1,4 ⟨R⟩
PRINT#1,CHR$(27) "0"; ⟨R⟩
        – sorközállítás
PRINT#1,CHR$(27) "C"CHR$(96); ⟨R⟩
        – laphossz sorban: 96
PRINT#1,CHR$(27) "N"CHR$(7);⟨R⟩
        – lapok közt kimaradó sor: 7
CMD1:LIST ⟨R⟩
– – – – – nyomtatás – – – –
PRINT#1:CLOSE1 ⟨R⟩
```

Az eredmény egészen elegáns! Programsoraink nem esnek hajtásra, az írás szebb, nyomtatott listánk tovább használható!

Egy javaslat:

Az apró gépi rutinok, azok futtatása sok örömet szerezhetnek az újság olvasóinak. Egy aprósággal még segíthetnének az olvasóknak: a basic betöltőben feltüntetnék, hogy a rutin stabil, vagy mobil. Azaz a rutin a memória más címein is futtatható-e?

Nagyobb használati értékűek az áthelyezhető rutinok, mert azok együtt is alkalmazhatóak; nem kell választani, hogy egyik vagy a másik!

Még jobb lenne, ha hozzáértő személy közlés előtt, átkomponálná a rutinokat mobillá!

●

Az 1992/5. sz. 29. lapon megjelent. Két program összefűzése hibás. A toldási módok keverednek, ismétlődnek.

A végrehajtás egyszerűbb:

1. Töltse be az első programot.
2. Állítsa át a basic terület kezdetét az alábbi sorokkal:

```
POKE43, (PEEK(45)) + 256*PEEK(46) — 2AND255 (return)
POKE44, (PEEK(45)) + 256*PEEK(46) — 2/256 (return)
```

3. Töltse be a második programot.
4. Állítsa vissza az eredeti kezdetet:

```
POKE43,1:POKE44,8 (return) és kész!
```

A közölt cikk csak akkor helyes, logikailag, ha az 1. és a 2. pontot felcseréljük!

Egyszerűbb azonban:

1. Töltse be az első programot.
2. Írja a képernyőre:

```
XX=PEEK(45) + PEEK(46)*256−2:PRINTXX(return)
```

3. A basic kezdet átállításához a kiírt értéket helyettesítse be:

```
POKE43,XX AND255:POKE44,XX/256(return)
```

4. Töltse be a második programot.
5. Állítsa vissza az eredeti basic kezdetet:

```
POKE43,1:POKE44,8
```

●

Egy ritkán alkalmazott programozási lehetőségre szeretnénk felhívni a lap olvasóinak figyelmét: a DATA-sorok REM-ekkel való ellátása. Nagy segítség lehet pl. karakterkészlet basic betöltőjében való eligazodásnál, ahol a megfelelő karakter a sor mögé írható.

A betöltőlista kinyomtatása után a karakterszerkesztővel megállapítható egy karakter helye; a DATA-sor végére kettőspontot téve beírható maga a karakter.

Ha a listán minden kívánt karakter értelmezett, a feljegyzés alapján a betöltő is átjavítható.

A DATA-sor különleges a maga nemében, itt a REM is különleges. Ezt az teszi lehetővé, hogy a DATA beolvasása után az értelmező vesszőt keres, ha nem azt talál, a következő sorra ugrik; „:"t írva már nem zavarók a feljegyzéseink!

A mellékelt programlista egy képernyőn használható ékezetes betűkészletet tartalmaz.

Tisztelettel

Mesterházi Sándor

```
1 REM EKEZETES KARAKTEREK KEPERNYORE
2 REM ========== NB / GR ============
6 REM SOFT:MESTERHAZI SANDOR-CELLDOMOLK
280 FORA=.TO48:READB:POKE12*4096+A,B:NEXT
290 DATA162,16,169,,141,14,220,169,51,133,1,169,208,160,,132,34,133
300 DATA35,169,112,132,36,133,37,177,34,145,36,200,208,249,230,35,230
310 DATA37,202,208,242,169,55,133,1,169,1,141,14,220,96
330 SYS49152
340 READA:IFA=-1THEN360
350 FORK=.TO7:READB:POKE28672+A*8+K,B:NEXTK:GOTO340
360 POKE53272,189:POKE56576,150:POKE648,108:PRINTCHR$(147):END
370 DATA249,231,231,195,153,153,153,195,255:O' RVS BE
371 DATA248,231,165,153,153,153,153,195,255:U'
372 DATA230,153,255,153,153,153,153,195,255:U..
373 DATA228,153,195,153,153,153,153,195,255:O..
374 DATA226,231,239,195,231,231,231,195,255:I'
375 DATA223,153,153,195,153,153,153,195,255:O''
376 DATA220,153,153,255,153,153,153,195,255:U''
377 DATA157,231,231,195,153,153,129,153,255:A'
378 DATA155,231,231,129,159,135,159,129,255:E'
379 DATA121, 24, 24, 60,102,102,102, 60,  0:O' <C=O> RVS KI
380 DATA120, 24, 90,102,102,102,102, 60,  0:U' <C=U>
381 DATA102,102,  0,102,102,102,102, 60,  0:U..<C=+>
382 DATA100,102, 60,102,102,102,102, 60,  0:O..<C=@>
383 DATA 98, 24, 16, 60, 24, 24, 24, 60,  0:I'<C=I>
384 DATA 95,102,102, 60,102,102,102, 60,  0:O''<C=*>
385 DATA 92,102,102,  0,102,102,102, 60,  0:U''<C=->
386 DATA 29, 24, 24, 60,102,102,126,102,  0:A' <C=]>
387 DATA 27, 24, 24,126, 96,120, 96,126,  0:E' <C=[>
389 DATA-1

READY.
```

Funkcióbillentyűk

A funkcióbillentyűk fontos szerepet játszanak a C 64 programozásában, bár nem részei a BASIC-nek. Legalábbis mostanáig nem. Az itt közölt program lehetővé teszi, hogy úgy programozzuk az F1 — F8 billentyűket, hogy a BASIC-ből adhassunk ki velük parancsokat.

Gépeljük be a programot és mentsük el. Futtatásakor a DATA sorokban lévő gépi kód a memóriába kerül és aktiválódik. Ezután NEW-val kitörölhetjük a programot: a funkcióbillentyűket kezelő gépi kód a C 64 kikapcsolásáig megmarad.

A programba beépített parancsok az alábbiak:

F1 – Háttér szin POKE-olása
F2 – Keret szin POKE-olása
F3 – Tartalomjegyzék betöltése
F4 – Képernyőtörlés
F5 – SAVE "
F6 – SYS
F7 – LIST
F8 – LOAD "

Mivel a gépi kód bent van a memóriában, a fenti parancsok megfelelő funkcióbillentyű lenyomásával kiadhatók.

Módosítás

A program könnyen adaptálható. A 20—90 sorok megfelelő átalakításával a különböző igényeknek megfelelően különböző változatokat hozhatunk létre. A 20-as sorban M$(1) vezérli az F1 billentyűt, a 30-asban M$(2) az F2-est, és így tovább.

Ha megváltoztatjuk ezeket a sorokat, ügyeljünk rá, hogy egyik parancs sem lehet hosszabb 8 karakternél, így célszerű lehet a rövidített alakok használata. Erre mutat példát a 20-as és 30-as sor, ahol P{SHIFT}/O szerepel POKE helyett.

Ha azonnal végrehajtható parancsokat szeretnénk, tegyünk +CHR$(13)-at a string végére. (lásd 40-es és 50-es sor) A gép a CHR$(13)-at RETURN-nek értelmezi. A hosszszámláláskor ez is egy karakternek számít. Hagyjuk el a RETURN-t az olyan parancsok végéről, melyek további információt igényelnek, mint például a SAVE, POKE vagy a SYS, illetve melyeket végrehajtás előtt nem árt mégegyszer ellenőrizni (pl. NEW). Az idézőjelet +CHR$(34)-el tehetjük be, mint a 60-as sorban.

A „Funkcióbillentyűk" program különösen akkor segít sokat, ha több gépi kódú programot használunk. Nem kell ilyenkor az összes SYS címet megjegyezni, rátehetjük őket a funkcióbillentyűkre. Például a 20-as sor így alakítható:

```
20 M$(1) = "S{SHIFT Y}49152"
```

Ezentúl ha BASIC-ben programozunk, ne tartsuk a funkcióbillentyűket haszontalannak. Inkább tekintsük szuperbillentyűknek, és fogjuk őket munkára.

```
100 REM FUNKCIO BILLENTYUK
110 :
120 M$(1)="P┌53281,"
130 M$(2)="P┌53280,"
140 M$(3)="L┌"+CHR$(34)+"$"+CHR$(34)+",8"+CHR$(13)
150 M$(4)="PRINT"+CHR$(147)+CHR$(13)
160 M$(5)="SAVE "+CHR$(34)
170 M$(6)="SYS "
180 M$(7)="LIST "+CHR$(13)
190 M$(8)="LOAD "+CHR$(34)
200 REM PARANCS KIIRAS
210 FOR I=1 TO 8:READ A:O(I)=A:NEXT
220 DATA 7,1,3,5,8,2,4,6
230 REM GEPIKODU PROGRAM
240 FOR I=53000 TO 53094:READ A:POKE I,A:NEXT I
250 REM POKE UZENET A MEMORIABOL
260 FOR J=1 TO 8
270 M$(O(J))=LEFT$(M$(O(J)),8)
280 T$=M$(O(J))
290 FOR I=I TO I+7
300 POKE I,ASC(T$+CHR$(0))
310 IF T$<>"" THEN T$=RIGHT$(T$,(LEN(T$)-1))
320 NEXT I:NEXT J
330 REM POKE UZENET A MEMORIABOL
340 FOR J=1 TO 8
350 POKE I,LEN(M$(O(J)))
360 I=I+1
370 NEXT J
380 SYS 53000:REM FUNKCIO BOLLENTYUK BEKAPCSOLASA
390 DATA 120,169,21,141,20,3,169,207,141,21,3,88,96,165,197,205,86,207
400 DATA 240,55,141,86,207,56,233,3,201,4,176,45,166,212,208,41,174,141
410 DATA 2,240,3,24,105,4,168,185,167,207,133,198,152,10,168,185,87,207
420 DATA 133,251,185,88,207,133,252,160,0,177,251,240,8,153,119,2,200
430 DATA 192,8,208,244,76,49,234,0,103,207,111,207,119,207,127,207,135
440 DATA 207,143,207,151,207,159,207

READY.
```

Fizikafeladat megoldása számítógéppel

A középiskolai Matematikai Lapok 2361. számú fizika feladata az alábbi volt:

Az azonos magasságban, egymástól 9 méterre lévő A és B pont között 13 m hosszú fonál van. A fonálra A-tól 5 méterre egy m=2 kg tömegű testet, B-től 5 méterre pedig egy M=3 kg tömegű testet akasztottunk.

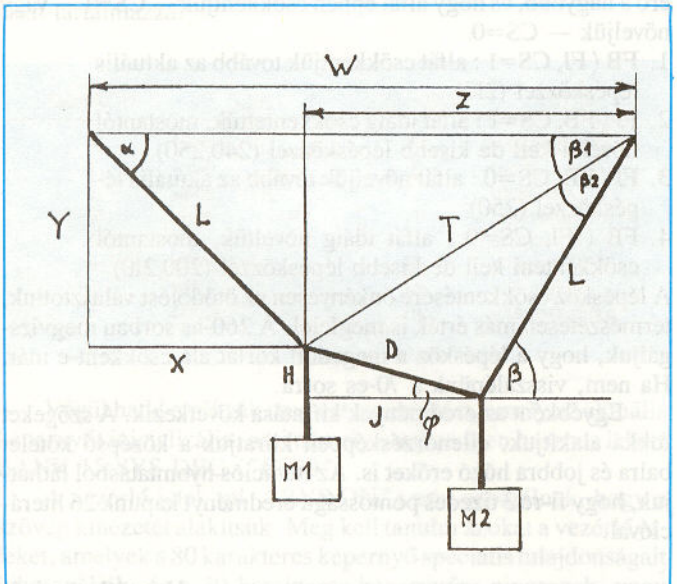
Határozzuk meg közelítőleg (1° pontossággal), hogy milyen szöget zár be a testeket összekötő fonál a vízszintessel!

A lapban közölt megoldás az ábrán látható távolságokra és szögekre felírt geometriai összefüggésekből valamint a kötéldarabok csatlakozópontjaiban ható erők egyensúlyából fi-re ezt az egyenletet hozta ki:

$$\operatorname{tg}\varphi = \frac{\frac{3}{5}\sqrt{1+\frac{24}{5}\cos\varphi-\frac{81}{25}\cos^2\varphi}+5\cdot\frac{\sin\varphi(3-\cos\varphi)}{5-3\cos\varphi}}{4-\frac{27}{5}\cos\varphi+5\cdot\frac{9-6\cos\varphi+\cos 2\varphi}{10-6\cos\varphi}}$$

Majd közölte, hogy az egyenlet numerikusan — táblázat segítségével vagy számítógéppel — megoldható.

Ha már úgysem adható explicit képlet a középső kötéldarab vízszintessel bezárt szögére, oldjuk meg valóban számítógéppel a feladatot. Képzeljük el, hogy a középső kötéldarab közepén van egy kis gyűrű, melynél fogva tetszőleges helyzetben rögzíthetjük a rendszert. Szimmetrikus helyzetben a jobboldali nehezebb súly miatt nyilván nagyobb erő húzza a gyűrűt jobbra, mint balra. Az ábrán alfával jelölt szög tetszőleges értékéhez kiszámíthatjuk az elrendezés többi geometriai adatát, ezekből pedig a középső kötéldarabot balra és jobbra húzó erőket.



Mozdítsuk el a gyűrűt jobbra kis lépésekben, például alfát fokonként csökkentve, ügyelve arra, hogy a kötéldarab egyenes maradjon. Egyszerre csak azt tapasztaljuk, hogy a gyűrűt már nagyobb erő húzza balra mint jobbra. Ekkor induljunk el visszafelé, de csökkentett lépésközzel. Néhány lépés után ismét a jobbra húzó erő lesz nagyobb. Most ismét csökkentjük a lépésközt és irányt váltunk. Mindezt addig folytatjuk, míg a lépésköz valamilyen előírt kis érték alá nem csökken.

```
10 REM *****************
20 REM *  FIZIKA PELDA  *
30 REM *****************
40 PI=ATN(1)*4
50 W=9:L=5:D=3:M1=2:M2=3:CS=1
     :DELT=PI/180:EPS=10-6
60 AR=ATN(4/3)
70 X=L*COS(AR):Y=L*SIN(AR):Z=W-X
80 T=SQR(Z*Z+Y*Y):B1=ATN(Y/Z)
90 CB2=(T*T+L*L-D*D)/(2*T*L)
100 B2=ATN(SQR(1-CB2*CB2)/CB2)
110 BR=B1+B2
120 H=L*SIN(BR)-Y:J=Z-L*COS(BR)
130 FR=ATN(H/J)
140 FB=M1*COS(AR)/SIN(AR-FR)
150 FJ=M2*COS(BR)/SIN(BR+FR)
160 FF=FR*180/PI
170 LPRINT AR,FF,FB,FJ,DELT
180 IF FJ<FB THEN 230
190 IF CS=1 THEN 210
200 DELT=DELT/5:CS=1
210 AR=AR-DELT
220 GOTO 260
230 IF CS=0 THEN 250
240 DELT=DELT/5:CS=0
250 AR=AR+DELT
260 IF DELT>EPS THEN 70
270 LPRINT
280 AF=AR*180/PI:FF=FR*180/PI
     :BF=BR*180/PI
290 LPRINT "alfa= ";AF
300 LPRINT "beta= ";BF
310 LPRINT "fi=   ";FF
320 LPRINT "fb=   ";FB
330 LPRINT "fj=   ";FJ
340 LPRINT
350 END
```

Ábránk jelöléseit alkalmazva a program elején beállítjuk az adatokat (W,L,D,M1,M2). Cs-1 jelöli, hogy a lépegetést alfa csökkentésével kezdjük, DELT pedig a kezdőlépésköz (1 fok) radiánban. EPS az a lépésköz, ahol abbahagyjuk az iterációt (50.

.9272952	0	1.5	2.25	1.745329E-02
.9098419	1.963317	1.598804	2.120887	1.745329E-02
.8923886	3.862421	1.708718	2.011524	1.745329E-02
.8749353	5.708766	1.831746	1.917894	1.745329E-02
.857482	7.511609	1.970418	1.837087	1.745329E-02
.8609727	7.154155	1.941289	1.852333	3.490659E-03
.8644634	6.795165	1.912883	1.868023	3.490659E-03
.8679541	6.434663	1.885174	1.884167	3.490659E-03
.8714448	6.072545	1.858137	1.900784	3.490659E-03
.8707466	6.145124	1.863493	1.897421	6.981318E-04
.8700484	6.2176	1.868873	1.894079	6.981318E-04
.8693503	6.290013	1.87428	1.890756	6.981318E-04
.8686522	6.362395	1.879715	1.887449	6.981318E-04
.867954	6.434663	1.885174	1.884167	6.981318E-04
.8680937	6.420233	1.884081	1.884821	1.396264E-04
.8680657	6.423105	1.884299	1.884691	2.792527E-05
.8680378	6.426002	1.884518	1.88456	2.792527E-05
.8680098	6.428908	1.884738	1.884427	2.792527E-05
.8680154	6.428327	1.884694	1.884453	5.585054E-06
.868021	6.427749	1.88465	1.88448	5.585054E-06
.8680266	6.427172	1.884606	1.884506	5.585054E-06
.8680322	6.426579	1.884562	1.884533	5.585054E-06
.8680378	6.425996	1.884518	1.88456	5.585054E-06
.8680367	6.426115	1.884527	1.884554	1.117011E-06
.8680356	6.426254	1.884537	1.884547	1.117011E-06
.8680344	6.426369	1.884545	1.884542	1.117011E-06

```
alfa= 49.73472
beta= 56.12091
fi=  6.426369
fb=  1.884545
fj=  1.884542
```

sor). Kiindulóhelyzetünkben az 5 méteres kötéldarabok vízszintes vetülete 3 méter, így két oldalt egy-egy 3—4—5 m oldalú háromszög szerepel, alfa kezdőértéke tehát arctg(4/3) (60-as sor).

A 70-es sortól indul az iteráció. Alfa aktuális értékéhez kiszámítjuk a baloldali kötéldarab függőleges és vízszintes vetületét, Y-t és X-et, majd az utóbbiból Z-t (70-es sor). T mint az Y és Z befogójú derékszögű háromszög átfogója adódik. Ennek a háromszögnek Y-nal szemközti szöge betal (Bl) (80-as sor).

A T,D,L oldalú háromszög D-vel szemközti B2 szögének CB2 koszinuszát a koszinusztétellel a 90-es sorban számítjuk ki. Mivel BASIC-ünkben nines arkusz koszinusz csak arkusz tanges függvény, magát a szöget úgy kapjuk meg, hogy tangensét kifejezzük a koszinuszával, majd ennek vesszük az arkusz tangensét (100-as sor). Béta (BR) B1 és B2 összege. Most már könnyen kiszámíthatjuk a középső kötéldarab függőleges és vízszintes vetületét (H és J) valamint a fi (FR) szöget. A szögek jelölésénél a programban az R betűvel a radián értékekre utalunk.

A középső kötéldarab két végpontjában ható erők vektorháromszögét felrajzolva láthatjuk, hogy ott alfa és béta pótszögei szerepelnek, így a kötelet húzó erőket a színusztétellel kiszámítva a 140-es és 150-es sor képleteit kapjuk. Hogy a program működését jobban nyomon követhessük, fi-t átszámítjuk fokra és kiírjuk a feladat néhány jellemző adatát.

Ezután következik az iteráció szervezése. A program valójában négyfelé ágazik aszerint, hogy a jobbra vagy a balra húzó erő a nagyobb, és hogy alfát éppen csökkentjük — CS=1 — vagy növeljük — CS=0.

1. FB 〈 FJ, CS=1 : alfát csökkentjük tovább az aktuális lépésközzel (210)
2. FJ 〈 FB, CS=1 : alfát idáig csökkentettük, mostantól növelni kell de kisebb lépésközzel (240,250)
3. FJ 〈 FB, CS=0 : alfát növeljük tovább az aktuális lépésközzel (250)
4. FB 〈 FJ, CS=0 : alfát idáig növeltük, mostantól csökkenteni kell de kisebb lépésközzel (200,210)

A lépésköz csökkentésére önkényesen az ötödölést választottuk, természetesen más érték is megfelel. A 260-as sorban megvizsgáljuk, hogy a lépésköz a megadott korlát alá csökkent-e már. Ha nem, visszalépünk a 70-es sorra.

Egyébként az eredmények kiíratása következik. A szögeket fokká alakítjuk, ellenőrzésképpen kiiratjuk a középső kötelet balra és jobbra húzó erőket is. Az iterációs nyomtatásból láthatjuk, hogy fi-re 3 tizedes pontosságú eredményt kapunk 26 literációval.

Horváth András

A C128-as szuperchipje

Mialatt a C128 legtöbb építőelemét már a C64-ből is ismerjük, azok kielégítően dokumentálva is vannak, a VDC, a 80 karakteres chip kissé titokzatosnak tűnik. Ez a cikk igyekszik segíteni abban, hogy jobban megismerjük ennek a chipnek a kezelését.

A 80 karakteres üzemmód képeiért a „Video Display Controller" (röviden VDC) a felelős, analóg módon a VIC-hez, amely tevékenységi köre a 40 jeles képernyőre terjed ki.

Hogy rögtön az elején leszögezzük: A VDC messzemenően nincs még kihasználva, és biztos csodálkozni fogunk, mi mindent is rejt még magában. Most a szöveges kivitel legegyszerűbb módozatával fogunk foglalkozni, méghozzá a PRINT paranccsal.

Szinte minden képernyős kivitelt a BASIC 7.0 PRINT vagy a CHAR utasításaival bonyolítunk le. Ha eleddig a CHAR parancsot csupán szövegeknek a grafikus módban történő megjelenítésére használtuk volna, akkor próbáljuk ki a következő parancsot: CHAR,5,10,"5. OSZLOP/10. SOR"

Ez az utasítás a kurzort a 10 sor 5. oszlopába viszi (ahol a számolás minden esetben az adott WINDOW bal fölső sarkától, a 0 pozíciótól kezdődik), és kiadja az idézőjelbe tett szöveget. Ha a sor végéhez egy ,1-et illesztünk, akkor a szöveg inverz módon fog megjelenni:

CHAR,5,10,"INVERZ SZÖVEG",1

Ha egy programból a 80 karakteres képernyőt szeretnénk kiválasztani, akkor a GRAPHIC 5 utasítást kell kiadni. Ha ilyenkor a ,1-et is hozzátesszük (GRAPHIC 5,1), akkor egyúttal töröljük is a képernyőt.

Az RWINDOW (2) paranccsal tudjuk azt megvizsgálni, melyik módban is van a C128 (40— vagy 80 karakteres üzemmód).

PRINT RWINDOW(2)

A COLOR parancs a képernyő színeinek beállítására szolgál. A COLOR 5, színkód a karakterek színét, a COLOR 6, színkód a háttér színét változtatja meg. A színkódokat az alábbi táblázat tartalmazza.

SZÍNKÓD	SZÍN	SZÍNKÓD	SZÍN
0	fekete	8	világosbarna
1	fehér	9	barna
2	piros	10	rózsaszín
3	zöld	11	sötétszürke
4	lila	12	szürke
5	sötétzöld	13	világoszöld
6	kék	14	világoskék
7	sárga	15	világosszürke

Végül hadd említsük meg azt az utasítást, amely az aktuális képernyőt inicializálja, ami némely esetben igen hasznos lehet:

BANK 15:SYS DEC ("C000")

A vezérlő jelek (pl. "RVS ON") arra szolgálnak, hogy a szöveg kinézetét alakítsuk. Meg kell tanulni azokat a vezérlő jeleket, amelyek a 80 karakteres képernyő speciális tulajdonságait kihasználják, és a 40 karakteres képernyőre nincsenek, vagy más hatással vannak. Ezeket a jeleket a PRINT utasítással kell kiadni. A PRINT CHR$(147) például törli a képernyőt.

A CHR$(2) olyan vezérlő jel, amelyet a parancsmódban a CTRL+B billentyű-kombinációval válthatunk ki, azaz bekapcsolja az aláhúzást. A CHR$(30) azt ismét kikapcsolja.

Mialatt a 40 karakteres üzemmódban a képernyőn vagy a kisbetű/nagybetű, vagy a negybetű/grafikus jelek módban dolgozhatunk, a 80 karakteres képernyőn minden egyes jelnél beállíthatjuk, az melyik jelkészletből legyen véve. A legjobb, ha ezt mindjárt kipróbáljuk a képernyőn. Állítsuk be a bekapcsolás utáni állapotot a reset gomb megnyomásával, majd adjunk be néhány grafikus karaktert (SHIFT+nagybetűk). Nyomjuk meg ezután a SHIFT és a C= billentyűt egyszerre, ahogy azt a C64 esetében szoktuk. Nem történik semmi? Akkor jó, mivel a jelkészlet váltása a már a képernyőn álló karaktereket nem szabad hogy befolyásolja. Adjunk be most további jeleket. Észre fogjuk venni, hogy most a kisbetű/nagybetű jelkészlet aktív. A CHR$(14)-gyel kapcsolhatunk át egyébként programból ebbe az üzemmódba, amelyet szöveg-jelkészletnek is neveznek. A nagybetű/grafikus jelek módra pedig a CHR$(142)-vel válthatunk. Ezt a módot azért nevezik így, mivel az összes grafikus jel a rendelkezésünkre áll, de csupán nagybetűket tudunk megjeleníttetni a képernyőn.

Attribut vezérlő jelek

A CHR$(15) segítségével az összes ezt követő jelet villogtatva adjuk ki. Ez nem is olyan haszontalan dolog, ha például fontos jelzéseket ílymódon szeretnénk kiemelni. Ugyanezt a hatást érhetjük el a CTRL+O billentyű-kombinációval is. A CHR$(143) kikapcsolja ezt a hatást.

A teljesség kedvéért hadd említsük meg azt, hogy ezeket a vezérlő jeleket egymással kombinálni is lehet. Tehát a villogó szövegeket akár aláhúzva is ábrázolhatjuk.

Az escape-szekvenciákat, ahogy azt a kézikönyv is írja a CHR$(27)''escape-utasítás'' formában, a PRINT paranccsal adhatjuk ki. Például a:

PRINT CHR$(27)''V'' képernyőscrollt (görgetést) vált ki fölfelé.

Álljanak itt most a mindenekelőtt vagy kizárólag a 80 karakteres képernyőre vonatkozó escape szekvenciák:

ESC R: Ebben az állapotban a teljes képernyőt invertáljuk. Műszakilag nézve ez semmilyen összefüggésben nincs a CHR$(18) és a CHR$(146) vezérlő jelekkel, amelyek az inverz nyomtatást kapcsolják be és ki egyes jelekre vonatkoztatva, bár a hatás hasonló.

ESC N: Ezzel a szekvenciával lehet az ESC R-t hatástalanítani.

ESC O: Ezzel minden üzemmódot (villogtatás, aláhúzás, inverz kijelzés) kikapcsolunk.

ESC U: A kurzort a 80 karakteres képernyőn aláhúzó kurzorként használhatjuk, ami meglehetősen profiként hat. Ezt a ''vonás'' kurzort bizonyos határok között még át is lehet definiálni.

ESC S: Ezzel az ESC U hatását kikapcsoljuk, a képernyőn ismét a normál blokk-kurzor jelenik meg.

VEZ. JELEK/ESC. SZEKVENCIÁK	HATÁS
CHR$(2)	aláhúzás be
CHR$(14)	kisbetű/nagybetű üzemmód
CHR$(15)	villogtatás be
CHR$(27)"N"	normál kijelzés (inverz ki)
CHR$(27)"O"	villogtatás/aláhúzás/inverz ki
CHR$(27)"R"	inverz ábrázolás
CHR$(27)"S"	a blokk-kurzor megjelenítése
CHR$(27)"U"	a vonáskurzor megjelenítése
CHR$(130	aláhúzás ki
CHR$(142)	nagybetű/grafikus jelkészlet
CHR$(143)	villogtatás ki

Legyen ez egyelőre elég kezdetnek. Mielőtt azonban elbúcsúznánk, szeretnénk összefoglalni, mi mindent kell a VDC-nek megjegyeznie ahhoz, hogy egy karaktert kijelezzen a képernyőn:

1. Melyik jelről is van szó (képernyőkód)?
2. Milyen színe van a jelnek (színkód)?
3. Inverz-e a kijelzés?
4. Alá van-e húzva?
5. Villog-e?
6. A kisbetű/nagybetű, vagy a nagybetű/grafikus jel jelkészletből van-e?

A 3—6. információkat egy jel attributjainak nevezzük; jó ha megjegyezzük magunknak ezt a fogalmat, mivel szükségünk lesz rá. A 40 karakteres képernyő csupán az 1. és 2. információt ismeri. A 3. a képernyőkódban van benne, hisz ha a kód) 127, akkor inverz lesz a kijelzés. A 6. pont pedig az egész képernyőre lesz mindenkor rögzítve. Az attribut tehát a 80 karakteres képernyő egy specialitása.

C-64 bővítések

PLOFI Univerzális autostartos EPROM-kártya 8/16 K-ra

Ez a kártyatípus a C-64 leggyakrabban és legsokoldalúbban használható bővítője.

A kártya kapcsolási rajza a *1. ábrán* látható. Ebben a 2764 és 27128 változtatás nélkül felhasználható. Ezután már csak az EPROM tartalma dönti el, hogy mit produkál a kártya.

A Hobbi Elektronikánál jelenleg az alábbi EPROM-ok kaphatók, amelyek az univerzális panelbe, önállóan feliratozott műanyag dobozba építhetők be:

1. Datassette cartridge-hez. Az EPROM tartalmaz ABC turbót, fejbeállítót, COPY 250-et, szalagcsévélőt, Turbo 250-et, kikapcsoló rutint.
2. EPROM-égető cartridge-hez. Tartalmazza az égető szoftvert.
3. HELP+cartridge. Tartalmazza a HELP+ programot, ami a programozó munkáját segíti.

A készlet változik, de a változtatást maga a felhasználó is elvégezheti.

Az égető szoftverje tartalmaz egy modulgenerátort, amivel szoftverismeret nélkül lehet autostartos kártyát készíteni. Ez azt jelenti, hogy ha van egy vagy több kedvenc programunk, amit nagyon sokszor használunk, akkor azt betölthetjük egy megfelelő kapacitású EPROM-ba. A lemezről behívott programhoz a modulgenerátor automatikusan hozzáfűzi az autostartos részt. Lehetőséget nyújt arra, hogy megszerkesszük a bejelentkező képet, több program esetén a menü formátumát. Mindezt lemezre menthetjük, ahonnan az égető menüje szerint beégethető az EPROM-ba. Az égető szoftvere arról is informál, ha az összeállított programcsomag tárigénye meghaladja az EPROM kapacitását, 8 K-t.

Ezután vizsgáljuk meg, hogy mi az autostartos kártya működésének alapja. Ez a kártyacsalád a kisfiamtól kapta a PLOFI becenevet.

A PLOFI működése

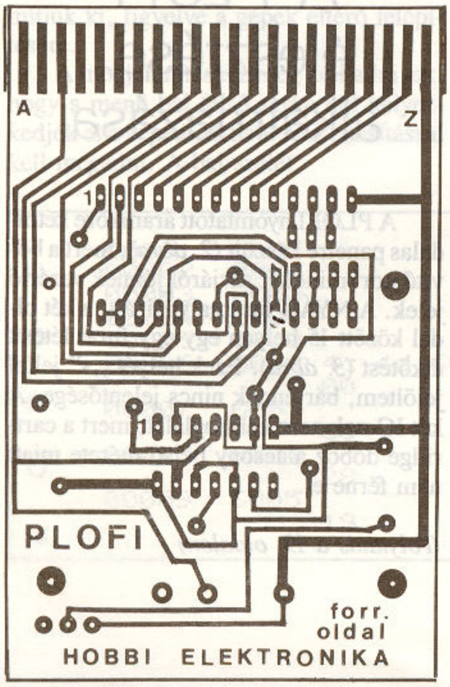
A C-64 RESET rutinja úgy van megírva, hogy bekapcsoláskor megvizsgálja: a $8004-es memóriahelytől kezdve a „CBM80" siftelt karakterei vannak-e (lásd *1. programrészlet*). Egyezés esetén a vezérlést a $8000-$8001 byte-ok által mutatott címre adja át. A $8002-8003-as byte-ok ugyanehhez a folyamathoz az NMI megszakítás esetére tartalmaznak címet.

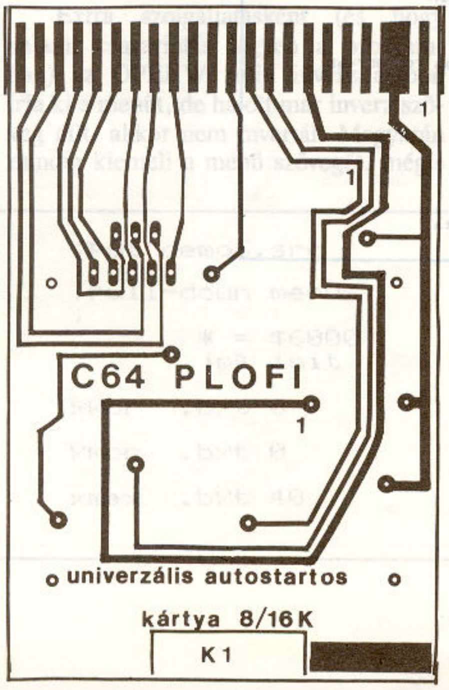
Ha egy olyan EPROM-ot helyezünk el $8000-től, amelyben az első 9 byte ilyen felépítésű, a gép bekapcsoláskor, vagy egy későbbi RESET, esetleg NMI megszakítás alkalmával nem a BASIC értelmezővel jelentkezik be, hanem egy általunk beégetett program indul el.

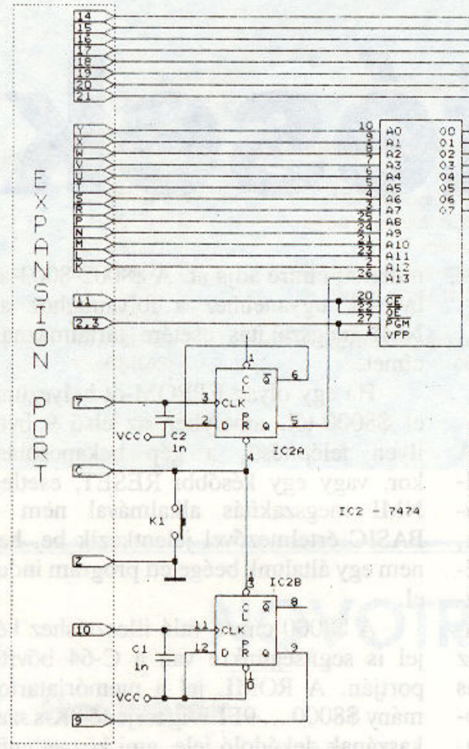
A $8000 címre való illesztéshez két jel is segítségünkre van a C-64 bővítő portján. A ROML jel a memóriatartomány $8000 ... 9FFF-ig terjedő 8K-s szakaszának dekódoló jele, ami közvetlenül felhasználható egy külső EPROM-chip engedélyezésére. Ezt a jelet a gépben található memória manager állítja elő. Ahhoz azonban, hogy ez a jel létrejöjjön, a memória manager EXROM bemenetét alacsony szintre kell húzni. Egyszerű felépítést tervezve, ezzel kész a feladat.

Az ilyen egyszerű megoldásnak több hibája van, ezért célszerű az EPROM tartalmát (lásd *2. programrészlet*) a gép RAM-jába átmásolni és a másolás után szoftver úton kikapcsolni, mintha ott sem lenne. Erre a műveletre az $\overline{I/O_1}$ vagy I/O_2 jel ad lehetőséget. Egy nagyobb kapacitású EPROM-ba (pl. 2764, 27128) több különféle segédprogramot is elhelyezhetünk. Ekkor célszerű egy címlistát elhelyezni a memóriaterület elején, hogy csak a kiválasztott program kerüljön beolvasásra.

Az 1. ábrán látható áramköri rajz ezt a megoldást biztosítja. Bekapcsoláskor vagy K_1 működtetésekor az IC_2 Q-ja alacsony szintre kerül és aktivizálódik az EXROM jel, a gép beolvassa az IC_1 elején elhelyezett menüt. Ebből kiválasztja a szükséges segédprogramot és beolvassa a RAM-ba. Amikor ez megtörtént I/O_2 átírja az IC_2 tartalmát, a Q H szintre kerül és bakapcsolja az IC_1-et. A K_1 RESET működtetésével ez mindig megismétlődik. Az IC_2 második fele 27128 esetén az A_{13} címet biztosítja a 8 K-s lapozáshoz. Órajelét I/O_1 biztosítja egy lapozó rutin segítségével. C_1 C_2 időkorrekciót végez. A tapasztalat szerint a régebbi kiadású gépeknél van rájuk szükség.





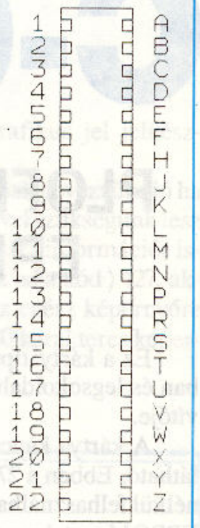


A PLOFI DATASET KAPCSOLÁSI RAJZA

EXPANSION PORT

Lab	Jeloles	Funkcio
1	GND	Rendszer fold
2	+5VDC	A felhasznaloi port es a CARTRIDGE
3	+5VDC	fogyasztasa nem haladja meg a 450mA-t
4	IRQ	Megszakitas kerelem
5	R/W	READ/WRITE
6	DOT CLOCK	8.18 MHz-es video orajel
7	I/O1	I/O blokk 1 ($DE00-$DEFF, meghajtas nelkuli)
8	GAME	TTl input
9	EXROM	TTl input
10	I/O2	I/O blokk 2 ($DF00-$DFFF, meghajtott LS TTL kimenet)
11	ROML	8K dekodolt RAM/ROM blokk ($8000, meghajtott LS TTL kimenet)
12	BA	Busz engedelyezese
13	DMA	Kozvetlen memoria eleresi kerelem
14	D7	Adat busz 7. bit
15	D6	Adat busz 6. bit
16	D5	Adat busz 5. bit
17	D4	Adat busz 4. bit
18	D3	Adat busz 3. bit
19	D2	Adat busz 2. bit
20	D1	Adat busz 1. bit
21	D0	Adat busz 0. bit
22	GND	Rendszer fold

Lab	Jeloles	Funkcio
A	GND	Rendszer fold
B	ROMH	8K dekodolt RAM/ROM blokk ($E000, meghajtott)
C	RESET	6502 Reset lab (meghajtott output/ meghajtas nelkuli input)
D	NMI	6502 nem maszkolhato megszakitas (meghajtott TTL output/meghajtas nelkuli input)
E	FI2	Rendszer orajel
F	A15	Cim busz 15. bit
H	A14	Cim busz 14. bit
J	A13	Cim busz 13. bit
K	A12	Cim busz 12. bit
L	A11	Cim busz 11. bit
M	A10	Cim busz 10. bit
N	A9	Cim busz 9. bit
P	A8	Cim busz 8. bit
R	A7	Cim busz 7. bit
S	A6	Cim busz 6. bit
T	A5	Cim busz 5. bit
U	A4	Cim busz 4. bit
V	A3	Cim busz 3. bit
W	A2	Cim busz 2. bit
X	A1	Cim busz 1. bit
Y	A0	Cim busz 0. bit
Z	GND	Rendszer fold



RESET RUTIN

1.Programreszlet

```
************************************
FCE2    A2 FF      LDX   #FF
FCE4    78         SEI
FCE5    9A         TXS
FCE6    D8         CLD
FCE7    20 02 FD   JSR   FD02    ellenorzo rutinra ugras
FCEA    D0 03      BNE   FCEF    nincs autostartos program?
FCEC    6C 00 80   JMP   (8000)  ugras az autostart cimre
FCEF    8E 16 D0   STX   D016    videovezerlo-2. vezerloregiszter
FCF2    20 A3 FD   JSR   FDA3    megszakitas elokeszitese
FCF5    20 50 FD   JSR   FD50    munkaterulet inicializalasa
FCF8    20 15 FD   JSR   FD15    vektorok beallitasa
FCFB    20 5B FF   JSR   FF5B    video-reset
FCFE    58         CLI
FCFF    6C 00 A0   JMP   (A000)  BASIC-hidegindítas
********************************  ellenorzo rutin
FD02    A2 05      LDX   #05
FD04    BD 0F FD   LDA   FD0F,X
FD07    DD 03 80   CMP   8003,X osszehasonlitas 'CBM80'-al
FD0A    D0 03      BNE   FD0F
FD0C    CA         DEX
FD0D    D0 F5      BNE   FD04
FD0F    60         RTS
********************************
FD10    C3 C2 CD C8 30              'CBM80' minta
```

EPROM TARTALOM

2.Program reszlet

```
*************************************
8000   09 80 09 80 C3 C2 CD 38 30 ugrasi cimek,'CBM80'karakterei
************************************* RESET rutin folytatasa
8009 8E 16 D0    STX   D016          videovezerlo-2. vezerloregiszter
800C 20 A3 FD    JSR   FDA3          megszakitas elokeszitese
800F 20 50 FD    JSR   FD50          munkaterulet inicializalasa
8012 20 15 FD    JSR   FD15          vektorok beallitasa
8015 20 5B FF    JSR   FF5B          video-reset
8018 58          CLI

*************************************
Menu rutin (tobb program eseten)
*************************************
Masolo rutin
*************************************
Program vagy programok
```

A PLOFI élesztése, alkalmazása

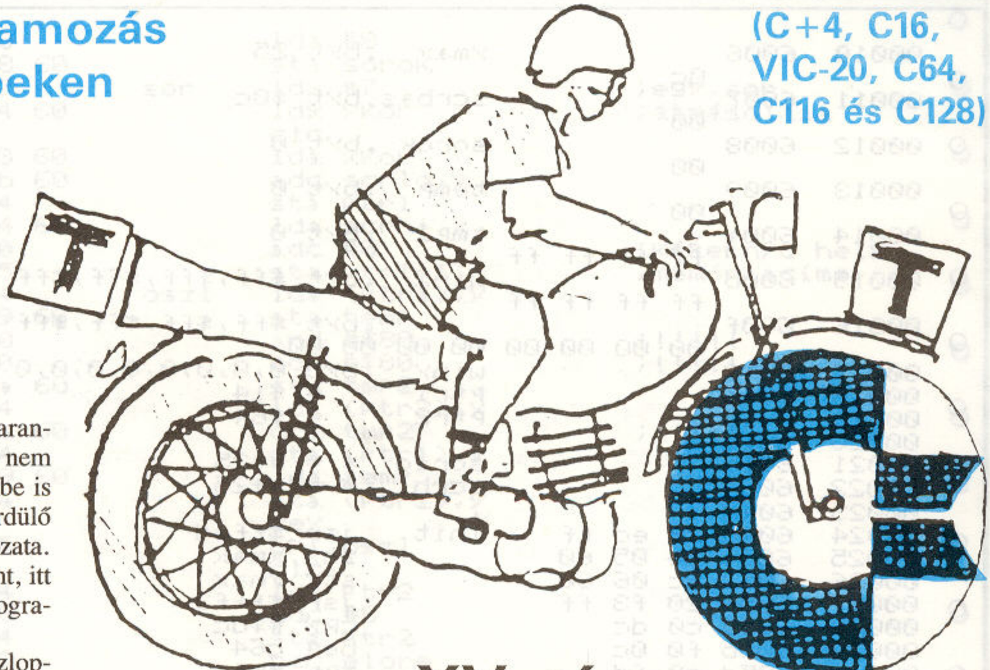
A PLOFI nyomtatott áramköre kétoldalas panelre készült (*2. ábra*), mert a bővítő port mindkét sávjáról jönnek vezérlő jelek. A NYÁK furatgalvanizált, a két oldal között 15 helyen egy-egy furat létesít átkötést (*3. ábra*). Ezek helyét „•" jellel jelöltem, bár ennek nincs jelentősége. A két IC-nek nem kell foglalat, mert a cartridge doboz alacsony belső mérete miatt nem férne el.

(Folytatás a 19. oldalon)

Gépi kódú programozás Commodore gépeken

(C+4, C16, VIC-20, C64, C116 és C128)

Menü tanuló módra



XV. rész

Utolsó példa programunk a beharangozott ablakkezelés helyett egy talán nem kevésbé érdekes, de a terjedelmünkbe is beleférő 'Pull-down menu' (legördülő menü...) rendszer kicsinyített változata. (Az ablak és menü szó bár mást jelent, itt keverni fogjuk. Ennek oka, hogy programunk a kettő között áll.)

Célunk 8 darab egyenként 8 oszlopból és 8 sorból álló, menü kezelése. Tudjuk a képernyő bármely pontján megjeleníteni ezeket, majd úgy eltüntetni, hogy az alattuk lévő képtartalom ne változzék meg. Mintha csak le- és felgördülne a menü szövege. Ehhez persze el kell tárolni a kiírandó menü helyén álló szöveget. Erre a legalkalmasabb terület a menü szövegének helye, magyarán kiíráskor (és persze majd visszaíráskor is) megcseréljük a képtartalmat a menü szövegével. Ez két egymásba ágyazott ciklussal történik. Az első lefutása után újra számoljuk a képernyőcímet, míg a menü adatban csak tovább lépünk a következő sorra (+8). Ezen eljárás miatt a menük szövege mindig 64 ⟨8*8⟩ byte legyen. Látható, hogy ezt már a menücím kiszámításnál is kihasználjuk, mert 64-gyel szorzunk és ahhoz adjuk a báziscímet.

A képernyősorok kezdetének memóriacímeit a korábbról ismert módon számítjuk ki, figyelve a gépek eltérő felépítésére.

A megjelenítéshez meg kell adni azt, hogy a menü bal felső sarka hol helyezkedjék el a képen. Ezt a POKE utasítással kell megadni. A bal szélét ⟨XKOR⟩ 24579 ⟨=$6000⟩-ra kell írni, a felső szélét ⟨YKOR⟩ 24580 ⟨=$6004⟩-re. Vigyázzunk, hogy a menü ne lóghasson ki a képből, mert a program a szükséges rövidség miatt nem ellenőrzi a paramétereket. Egyetlen ellenőrzés az, hogy csak megnyitott (a képernyőre kiírt) menüt hajlandó lezárni, és nem nyit meg újra már kirakott menüt. A képernyőn való átfedéskor fordított sorrendben kell becsukni (visszaíratni) a menüket. Ellenkező esetben az elmentett képernyőadatok (amik a menük alatt voltak) és a menük szövege összekeveredhet. A menü számát nyitáskor a $608f, becsukáskor $610d címre kell írni. Menüt nyitni a SYS24718 ⟨=$608e⟩-vel lehet. A becsukás SYS24844 ⟨—$610c⟩.

Extra szolgáltatásként (és hogy valami csavarintás legyen a programban) az OPENW rutin inverz alapon írja ki a menüt, de ha ott már inverz szöveg állt, akkor nem invertál. Magyarán mindig kiemeli a menü szövegét, mégis jól állítja vissza a CLOSEW rutin szöveget.

Az egész program megértéséhez tanácsoljuk, hogy próbálkozzon a Tisztelt Olvasó bátran, nyisson meg több ablakot, csukja le őket rossz sorrendben. Ekkor látni fogja, mely memória területeket használjuk és ott mit tárolunk. Érdemes a 94. sorral kezdődő bit-matatásba belenyúlni, megérthető, mikor inverz a kiírt szöveg és mikor nem.

Programunk fut minden szokásos gépen, kivétel a C-128, ahol csak 64-es üzemmódban. Az ok ami miatt sokszor tettük ezt a kitételt, a C-128 80 karakteres képernyőjének sajátos felépítése. Nem érhető el közvetlenül memóriaként, így teljesen újra kellett volna írni hozzá legtöbb programunkat. Ezek elvesztették volna példa jellegüket, mert a főprogram eltűnt volna a bonyolult körítésben.

Ezzel lezárjuk sorozatunkat, de továbbra is várjuk Olvasóink leveleit, melyekre ha szükséges, az újságban, cikk formájában adunk választ.

```
00001  0000                ;put"demo6.src
00002  0000                ;
00003  0000                ;pull-down menus
00004  0000                ;
00005  0000                      * = $6000
00006  6000 4c 4d 60             jmp init
            00
00007  6003                xkor  .byt 0          ;bal szél
            00
00008  6004                ykor  .byt 0          ;felső sor
            28
00009  6005                xmax  .byt 40         ;kep szélesség
            19
```

```
6006                     ymax   .byt 25                    ;kep magassag
      0c
6007                     scrbas.byt $0c                    ;kezdőcím/256
      00
6008                     sorok  .byt 0
      00
6009                     temp   .byt 0
      00
600a                     tmp2   .byt 0
      ff ff ff ff
600b                     winx   .byt $ff,$ff,$ff,$ff
      ff ff ff ff
600f                            .byt $ff,$ff,$ff,$ff
      00 00 00 00 00 00 00 00
6013                     winy   .byt 0,0,0,0,0,0,0,0
601b                     ptr1      = $14
601b                     ptr2   .  = $64
601b                     ;
601b                     scrlo   * = *+25
6034                     scrhi   * = *+25
604d                     ;
604d 20 ed ff            init   jsr $ffed                  ;screen
6050 8e 05 60                   stx xmax
6053 8c 06 60                   sty ymax
6056 20 f3 ff                   jsr $fff3                  ;iobase
6059 c0 dc                      cpy #$dc
605b f0 0c                      beq c64
605d c0 fd                      cpy #$fd
605f f0 04                      beq plus4
6061 a9 1e               vc20   lda #$1e                   ;vc20 bázis
6063 d0 06                      bne bazis
6065 a9 0c               plus4  lda #$0c                   ;c-16,plus/4
6067 d0 02                      bne bazis
6069 a9 04               c64    lda #$04                   ;c-64
606b 8d 07 60            bazis  sta scrbas
606e aa                         tax
606f a0 00                      ldy #0
6071 a9 00                      lda #0
6073 99 1b 60            szamol sta scrlo,y                ;képernyő sorok
6076 8d 09 60                   sta temp                   ;kezdőcímeinek
6079 8a                         txa                        ;táblázatát
607a 99 34 60                   sta scrhi,y                ;készíti el
607d ad 09 60                   lda temp
6080 18                         clc
6081 6d 05 60                   adc xmax
6084 90 01                      bcc szamel
6086 e8                         inx
6087 c8                  szamel iny
6088 cc 06 60                   cpy ymax
608b d0 e6                      bne szamol
608d 60                         rts
608e                     ;
608e a9 00               openw  lda #0                     ;ablakszám
6090 29 07                      and #7                     ;0-7-ig érvényes
6092 48                         pha
6093 aa                         tax
6094 bd 0b 60                   lda winx,x
6097 c9 ff                      cmp #$ff                   ;szabad-e
6099 d0 6f                      bne error                  ; az ablak ?
609b ad 03 60                   lda xkor
609e 9d 0b 60                   sta winx,x                 ;mostani helyzet
60a1 ad 04 60                   lda ykor                   ; eltárolása
60a4 9d 13 60                   sta winy,x
60a7 a9 00               masol  lda #0
60a9 85 64                      sta ptr2
60ab 68                         pla
60ac 4a                         lsr a                      ;ablakszám
60ad 66 64                      ror ptr2                   ;szorzása
60af 4a                         lsr a                      ;64-gyel
60b0 66 64                      ror ptr2
60b2 85 65                      sta ptr2+1
60b4 18                         clc                        ;plusz az
60b5 a5 64                      lda ptr2                   ;ablakok kezdete
60b7 69 2a                      adc #<winmap               ;megadja
60b9 85 64                      sta ptr2                   ;hol van az
60bb a5 65                      lda ptr2+1                 ;ablak adata
60bd 69 61                      adc #>winmap               ;a memóriában
60bf 85 65                      sta ptr2+1
```

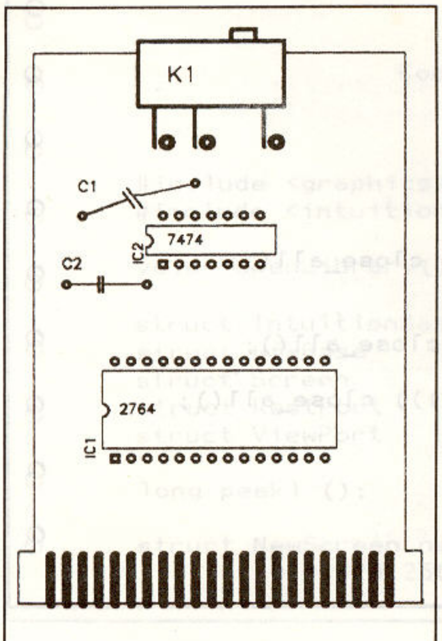
```
60c1 a9 08                    lda #8
60c3 8d 08 60                 sta sorok
60c6 a0 07          sor       ldy #7              ;egy sor
60c8 ae 04 60                 ldx ykor            ;átmásolása
60cb 18                       clc
60cc ad 03 60                 lda xkor
60cf 7d 1b 60                 adc scrlo,x
60d2 85 14                    sta ptr1
60d4 bd 34 60                 lda scrhi,x
60d7 69 00                    adc #0              ;képernyő hely
60d9 85 15                    sta ptr1+1          ;memóriacíme
60db b1 14          oszl      lda (ptr1),y
60dd 8d 09 60                 sta temp
60e0 29 80                    and #$80            ;!!!!!
60e2 49 80                    eor #$80            ;!!!!!
60e4 8d 0a 60                 sta tmp2
60e7 b1 64                    lda (ptr2),y
60e9 0d 0a 60                 ora tmp2            ;!!!!!
60ec 91 14                    sta (ptr1),y
60ee ad 09 60                 lda temp
60f1 91 64                    sta (ptr2),y
60f3 88                       dey
60f4 10 e5                    bpl oszl
60f6 18                       clc
60f7 a5 64                    lda ptr2
60f9 69 08                    adc #8
60fb 85 64                    sta ptr2
60fd 90 02                    bcc elore
60ff e6 65                    inc ptr2+1
6101 ee 04 60       elore     inc ykor
6104 ce 08 60                 dec sorok
6107 d0 bd                    bne sor
6109 60                       rts
610a                ;
610a 68             error     pla                 ;hibánál
610b 60                       rts                 ;visszatér
610c                ;
610c a9 00          closew    lda #0              ;ablakszám
610e 29 07                    and #7
6110 48                       pha
6111 aa                       tax
6112 bd 0b 60                 lda winx,x
6115 c9 ff                    cmp #$ff            ;foglalt-e
6117 f0 f1                    beq error           ;az ablak ?
6119 8d 03 60                 sta xkor
611c bd 13 60                 lda winy,x          ;visszaállítja
611f 8d 04 60                 sta ykor            ;az elmentett
6122 a9 ff                    lda #$ff            ;paramétereket
6124 9d 0b 60                 sta winx,x          ;   és
6127 4c a7 60                 jmp masol           ;visszamásol
612a                ;
612a                winmap * = *
     20 49 4d 45 20 45 5a 20
612a                .byt ' ime ez '           ;itt tökmindegy
     20 49 54 54 20 41 5a 20
6132                .byt ' itt az '           ;mi van, csak
     20 20 45 4c 53 4f 20 20
613a                .byt '  elso  '           ;8 darab ablak
     41 42 4c 41 4b 4f 4d 2e
6142                .byt 'ablakom.'           ;legyen, amely
     20 20 20 20 20 20 20 20
614a                .byt '        '           ;8*8 betűből
     20 4b 49 53 53 45 20 20
6152                .byt ' kisse  '           ;áll.
     20 54 41 4c 41 4e 20 20
615a                .byt ' talan  '
     53 5a 55 4b 4f 53 2e 20
6162                .byt 'szukos. '
616a                ;
     20 45 5a 20 20 41 20 20
616a                .byt ' ez  a  '
     4d 41 53 4f 44 49 4b 20
6172                .byt 'masodik '
     20 20 20 53 45 4d 20 20
617a                .byt '   sem  '
     53 5a 45 4c 45 2d 20 20
6182                .byt 'szele-  '
     20 20 20 20 53 45 42 42
```

```
00146   618a                              .byt '    sebb'
                20 4e 41 4c 41 20 20 20
00147   6192                              .byt ' nala    '
                20 20 20 20 20 20 20 20
00148   619a                              .byt '         '
                20 20 20 20 20 20 20 20
00149   61a2                              .byt '         '
00150   61aa                   ;
                20 33 2e 20 20 20 20 20
00151   61aa                              .byt ' 3.      '
                41 20 4d 45 52 45 54 20
00152   61b2                              .byt 'a meret  '
                4b 4f 54 4f 54 54 2c 20
00153   61ba                              .byt 'kotott,  '
                44 45 20 45 5a 20 20 20
00154   61c2                              .byt 'de ez    '
                4d 45 4e 55 4e 45 4c 20
00155   61ca                              .byt 'menunel  '
                4e 45 4d 20 20 20 20 20
00156   61d2                              .byt 'nem      '
                20 4c 45 48 45 54 20 20
00157   61da                              .byt ' lehet   '
                20 20 47 4f 4e 44 20 20
00158   61e2                              .byt '  gond   '
00159   61ea                   ;
                34 2e 4d 45 4e 55 20 20
00160   61ea                              .byt '4.menu   '
                4c 49 53 54 41 5a 41 53
00161   61f2                              .byt 'listazas'
                42 45 49 52 41 53 20 20
00162   61fa                              .byt 'beiras   '
                4b 49 4c 45 50 45 53 20
00163   6202                              .byt 'kilepes  '
                4a 41 56 49 54 41 53 20
00164   620a                              .byt 'javitas  '
                54 4f 52 44 45 4c 45 53
00165   6212                              .byt 'tordeles'
                54 4f 52 4c 45 53 20 20
00166   621a                              .byt 'torles   '
                52 4f 47 5a 49 54 45 53
00167   6222                              .byt 'rogzites'
00168   622a                   ;
                35 2e 20 20 20 20 20 20
00169   622a                              .byt '5.       '
                20 20 20 20 20 20 20 20
00170   6232                              .byt '         '
                20 20 20 20 20 20 20 20
00171   623a                              .byt '         '
                20 20 20 20 20 20 20 20
00172   6242                              .byt '         '
                20 20 20 20 20 20 20 20
00173   624a                              .byt '         '
                20 20 20 20 20 20 20 20
00174   6252                              .byt '         '
                20 20 20 20 20 20 20 20
00175   625a                              .byt '         '
                20 20 20 20 20 20 20 20
00176   6262                              .byt '         '
00177   626a                   ;
                36 2e 20 20 20 20 20 20
00178   626a                              .byt '6.       '
                20 20 20 20 20 20 20 20
00179   6272                              .byt '         '
                20 20 20 20 20 20 20 20
00180   627a                              .byt '         '
                20 20 20 20 20 20 20 20
00181   6282                              .byt '         '
                20 20 20 20 20 20 20 20
00182   628a                              .byt '         '
                20 20 20 20 20 20 20 20
00183   6292                              .byt '         '
                20 20 20 20 20 20 20 20
00184   629a                              .byt '         '
                20 20 20 20 20 20 20 20
00185   62a2                              .byt '         '
00186   62aa                   ;
                37 2e 20 20 20 20 20 20
```

```
00187  62aa                       .byt '7.         '
             20 20 20 20 20 20 20 20
00188  62b2                       .byt '           '
             20 20 20 20 20 20 20 20
00189  62ba                       .byt '           '
             20 20 20 20 20 20 20 20
00190  62c2                       .byt '           '
             20 20 20 20 20 20 20 20
00191  62ca                       .byt '           '
             20 20 20 20 20 20 20 20
00192  62d2                       .byt '           '
             20 20 20 20 20 20 20 20
00193  62da                       .byt '           '
             20 20 20 20 20 20 20 20
00194  62e2                       .byt '           '
00195  62ea                 ;
             38 2e 20 20 20 20 20 20
00196  62ea                       .byt '8. no ez'
             20 20 20 20 20 20 20 20
00197  62f2                       .byt 'vegre az'
             20 20 20 20 20 20 20 20
00198  62fa                       .byt ' utolso '
             20 20 20 20 20 20 20 20
00199  6302                       .byt ' 9 8 7  '
             20 20 20 20 20 20 20 20
00200  630a                       .byt '  6  5  '
             20 20 20 20 20 20 20 20
00201  6312                       .byt '  4  3  '
             20 20 20 20 20 20 20 20
00202  631a                       .byt '  2  1  '
             20 20 20 20 20 20 20 20
00203  6322                       .byt '  zero  '

end of assembly, error count = 00000

bazis    606b    c64      6069    closew   610c    elore    6101
error    610a    init     604d    masol    60a7    openw    608e
oszl     60db    plus4    6065    ptr1     0014    ptr2     0064
scrbas   6007    scrhi    6034    scrlo    601b    sor      60c6
sorok    6008    szamel   6087    szamol   6073    temp     6009
tmp2     600a    vc20     6061    winmap   612a    winx     600b
winy     6013    xkor     6003    xmax     6005    ykor     6004
ymax     6006
```



DATASET cartridge beultetési rajza

(Folytatás a 14. oldalról)

Az áramkört a C-64 kímélése miatt csak dobozolva szabad használni! A doboz mérete egyértelműen meghatározza a panel méretét. A doboz belmérete picit keskenyebb, mint a C-64 csatlakozójának szélessége, ezért a panelt a rajta levő vonal mentén úgy kell kivágni és megreszelni, hogy a 3. ábrán levő formát kapjuk. Ez kényelmesen elfér a dobozban és pontosan illeszkedik a csatlakozóhoz.

A panelt erős fénnyel átvilágítva, mindkét oldalt vizsgáljuk meg, hogy nincs-e rajta rövidzár vagy szakadás, esetleg rossz forrasztás. A gyanús pontot ohmmérővel ellenőrizzük. Forrasszuk be a két IC-t. A 2. ábrának megfelelően ellenőrizzük ohmmérővel a csatlakozósáv és az IC lábak közötti vezetékeket. Forrasszuk K_1-et ellenálláslábból készült forrcsúcsokhoz úgy, hogy a dobozból csak a működtetéshez szükséges méret lógjon ki. A panelen levő 3 db helyezőfurat segítségével dobozoljuk be az áramkört, ezzel az kész, rádughatjuk a gépre.

Figyelem! A C-64 bővítő portjának nincs vezetőcsapja, így a panelt fordítva is lehet csatlakoztatni. Ez azonban végzetes hibát okozhat, tönkreteheti a PLOFI-t vagy a gépet. A PLOFI helyesen ültetési felületével, az alkatrészekkel felfelé csatlakozik a géphez! A gépbe feltenni vagy levenni *csak kikapcsolt* állapotban szabad! A helyes pozíciót a doboz felirata mutatja. Erről a félről le kell vágni egy helyezőcsapot, hogy a panelhez illeszkedjen.

EGÉRxy

Ez a program a RUN utasítással elindítva, egy igen hasznos segítőtársunk lehet, ugyanis egy mozgatható ablakban megjeleníti az egér aktuális koordinátáit.

AMIGA

```
/*
          RUN EGERxy


             AztecC:   cc +l <file>
                       ln <file>.o -lc32 -lm -o

                  Code by TPE

                                             */

#include <functions.h>
#include <exec/types.h>
#include <intuition/intuition.h>
#include <intuition/intuitionbase.h>

struct IntuitionBase      *IntuitionBase;
struct GfxBase            *GfxBase;
struct RastPort           *RP;
struct IntuiMessage       *mes;
struct Window             *nwin,*awin;

struct NewWindow wind =
       {400,20,140,26,3,1,CLOSEWINDOW,WINDOWCLOSE|WINDOWDEPTH|WINDOWDRAG,
        NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,WBENCHSCREEN };

VOID close_all()
{
       if(nwin) CloseWindow(nwin);
       if(GfxBase) CloseLibrary(GfxBase);
       if(IntuitionBase) CloseLibrary(IntuitionBase);
       exit(0);
}

VOID main()
{
       int      x,y;
       char     line[11];
       ULONG    MessageClass;
       USHORT   code;

       if(!(IntuitionBase=(struct IntuitionBase *)
              OpenLibrary("intuition.library",0))) close_all();

       if(!(GfxBase=(struct GfxBase *)
              OpenLibrary("graphics.library",0))) close_all();

       if(!(nwin=(struct Window *) OpenWindow(&wind))) close_all();

       RP = nwin->RPort;

       for(;;)
       {
```

```
        awin = IntuitionBase->ActiveWindow;

        x = awin->MouseX;
        y = awin->MouseY;

        sprintf(line,"%4d ; %4d",x,y);
        Move(RP,25,20);
        Text(RP,&line,11);

        if(mes = (struct IntuiMessage *) GetMsg(nwin->UserPort))
        {
                MessageClass = mes->Class;
                code = mes->Code;

                ReplyMsg(mes);

                if(MessageClass == CLOSEWINDOW) close_all();
        }
    }
}
```

AMIGA

STARS

Igy ennek a programnak tulajdonképpen nincsen semmiféle gyakorlati haszna, de átböngészésével igen sok hasznos dolgot lehet belőle kiszűrni.

Bizonyára mindenki látott már olyan effektet, amely olyan hatást igyekszik kelteni, mintha egy űrhajóban utaznánk és csillagok úsznak el mellettünk.

Ez a programocska is ezt próbálja utánozni, csak nem a megszokott (assembly) programozási nyelven.

```
    /*
            STARS

                AztecC:  cc +l <file>
                         ln <file>.o -lc32 -lm -o

                    Code by TPE
                                                            */

#include <graphics/gfxbase.h>
#include <intuition/intuition.h>

void *OpenLibrary();

struct IntuitionBase        *IntuitionBase;
struct GfxBase              *GfxBase;
struct Screen               *scr, *OpenScreen();
struct RastPort             *rp;
struct ViewPort             *vp;

long peekl ();

struct NewScreen ns={
        0,0,320,256,5,0,0,NULL,CUSTOMSCREEN,NULL,NULL,NULL,NULL };
```

```
#define MAXSTARS 80

short lastc[MAXSTARS][2];
short reac [MAXSTARS][3];
short count[MAXSTARS];

short rnd()
{
        register long random;
        random = random<<2 ^ peekw(0xdff006) + 17910192L;
        return random & 511;
}

main()
{
        register short xc,yc,s;
        short col,i;

        IntuitionBase = OpenLibrary("intuition.library",0L);
        GfxBase = OpenLibrary("graphics.library",0L);

        scr = OpenScreen(&ns);

        rp = &scr->RastPort;vp = &scr->ViewPort;

        for (i=0; i<16; i++) SetRGB4(vp,(long)i,(long)i,(long)i,(long)i);

        for (i=0; i<MAXSTARS; i++)
        {
                reac[i][0] = rnd()+rnd()-512;
                reac[i][1] = rnd()+rnd()-512;
                reac[i][2] = rnd()>>1;
        }
        while (peek(0xbfec01) !=57)
        {
                for (s = 0; s< MAXSTARS; s++)
                {
                        if (reac[s][2] <= 0)
                        {
                                reac[s][0] = rnd()+rnd()-512;
                                reac[s][1] = rnd()+rnd()-512;
                                reac[s][2] = rnd()>>1;
                        }

                        else
                        {
                                xc = 40*reac[s][0]/reac[s][2]+160;
                                yc = 40*reac[s][1]/reac[s][2]+128;

                                col = 17-reac[s][2]/15;
                                if (col > 15) col = 15;

                                SetAPen(rp,0L);
                                WritePixel(rp,(long)lastc[s][0],
                                        (long)lastc[s][1]);

                                if (xc>=0 && yc>=0 && xc <=319 && yc<=255)
                                {
                                        SetAPen(rp,(long)col);
                                        WritePixel(rp,(long)xc,(long)yc);

                                        lastc[s][0] = xc;
                                        lastc[s][1] = yc;
                                }
```

```
        reac[s][2] -= 5;
     }
   }
   WaitTOF(vp);
 }
 CloseScreen(scr);
 CloseLibrary(IntuitionBase);
 CloseLibrary(GfxBase);
}
```

AMIGA

MaxWin

Bizonyára mindenkivel előfordult már olyan eset, hogy egyes felhasználásoknál az ablak méretet örökösen maximális méretre kellett „kihúzni" Ez igen bosszantó dolog, ha sűrűn fordul elő.

Ezen a kellemetlenségen segít a MaxWin nevű program. A startup-sequence-be iktatva a lemez indulásakor a CLI ablakát maximális méretűre tágítja ki, automatikusan megnövelve a munkaterület nagyságát.

```
/*
     MaxWin

          AztecC:  cc +l <file>
                   ln <file>.o -lc32 -lm -o

                    Code by TPE
                                                  */

#include <functions.h>
#include <exec/types.h>
#include <intuition/intuition.h>
#include <intuition/intuitionbase.h>

struct IntuitionBase     *IntuitionBase;
struct Window             *awin;

main()
{
     int       h,y;

     if(!(IntuitionBase=(struct IntuitionBase *)
          OpenLibrary("intuition.library",0))) exit(FALSE);

     awin = IntuitionBase->ActiveWindow;

     h = awin->Height;
     y = 256-h;

     if(h == 0) exit();

     SizeWindow(awin,0,y);

     CloseLibrary(IntuitionBase);
}
```

Számítógépes fogalmak

Lexikon

VIII. rész

Grafika

A C64 rendes körülmények között szöveges módban dolgozik. Itt 25 sorban soronként 40 jelet, azaz összesen ezret tudunk földolgozni, ábrázolni. A C64-es ebben a módban nemcsak betűket és számokat, hanem úgynevezett grafikus karaktereket, azaz blokkgrafikát is meg tud jeleníteni. Így lehetőség van bizonyos grafikus ábrázolásra, de igen korlátozott keretek között. Profi munkaként ható grafikákhoz a C64 a multicolor és a HiRes, azaz nagyfelbontású módokkal is rendelkezik. Ezek fölbontása 160×200, illetve 320×200 pixel, azaz képpont.

Van rá lehetőségünk, hogy ilyen nagyobb felbontású képeket jelenítsünk meg a képernyőn (és persze nyomtatón is), viszont ezek a képek jóval több helyet foglalnak el a tárolóban, mint a szöveges képernyő.

A nagyobb számítógépek esetében, mivel azok több tárolóhellyel rendelkeznek, nincs szükség szöveges tárolóra, azok ugyanis a karaktereket pontgrafikával is meg tudják jeleníteni.

CAD (Computer Aided Design)

Azaz magyarul: számítógépes támogatású tervezés. A CAD rendszerek olyan számítógépes rendszerek, amelyek a fejlesztőnek, tervezőknek segítséget nyújtanak szerszámok, épületek, kapcsolások stb. tervezésében. A papírt a képernyő helyettesíti.

Egy CAD munkahelyen a szokásos perifériák (képernyő, floppy, billentyűzet, nyomtató) mellett nagyfelbontású grafikus displayt, digitalizáló készüléket, fényceruzát vagy egeret is találhatunk. Ezekkel lehet a grafikus elemeket alakítani, módosítani, tervezni. A számítógépes dialógus során keletkező konstrukciót azután igen nagymértékben lehet alakítani, módosítani.

A CAD előnyei között szerepel többek között a tervek problémamentes változtathatósága. A rendszer hatékonyságától függően a CAD szimulációkra is alkalmas.

Animáció

Animáció alatt a grafikák, grafikus részletek vagy egyéb grafikus objektumok (pl. sprite-ok, shape-ek) mozgatását értjük.

A C64 esetében igen egyszerű spriteokat mozgatni, mivel bármiféle programozástechnikai trükk nélkül maximum 8 ilyen objektumot tudunk a háttértől teljesen függetlenül mozgatni. Tipikus alkalmazásnak a számítógépes játékok mozgó alakjait hozhatjuk föl.

Lehetőség van azonban rajzfilmek létrehozására a grafikus animációval. Minden trükkfilm olyan állóképekből áll, amelyeket azután a szemünk számára érzékelhetetlenül, legalább 24 kép/másodperces frekvenciával pergetnek le. A remegésmentesség ebben az esetben azt jelenti, hogy kihasználva a szemünk tehetetlenségét mi az állóképeket úgy érzékeljük, mintha az folyamatos mozgás lenne.

Bitmap

A 8 bites számítógépek leggyakrabban alkalmazott grafika technikája az úgynevezett bitmapping. Ez azt jelenti, hogy minden a képernyőn megjeleníthető képponthoz egy bit van hozzárendelve a tárolóban. Ha az adott bit értéke nulla, a hozzárendelt képpont ki-, ha viszont a megfelelő érték egy, akkor be van kapcsolva. Ha a C64-es bitmap üzemmódban van, akkor a felhasználó akár egyszerű BASIC parancsokkal (POKE) ki-bekapcsolhatja a képpontokat. A C64-es maximum 7 bitmapot igazgathat. Minden egyes pixel 1 bitet igényel a szabad RAM-ban. A 320×200 pixeles felbontásnál 64000 bitet, azaz 8000 byte-ot foglalunk el. Ha az összes elméletileg lehetséges bitmapot fölhasználjuk, akkor a 64 kbyte RAM-ból 7×8, azaz 56 kbyte területet foglalnánk le.

A C64-es bitmap területei az alábbi helyeken találhatók:
$2000, $4000, $6000, $8000, $A000, $C000, $E000

Algoritmus

Algoritmusnak nevezzük a tetszőleges problémák numerikus eljárással történő megoldásának tervét. Ilyen eljárásokat az idők során különböző szerzők dolgoztak ki, aholis különböző problémákhoz különböző algoritmusok mutatkoznak legmegfelelőbbnek. A grafikus felhasználások igen gyakran járnak komplex matematikai feladatokkal. Vonalhúzás, körrajzolás és egyebek, mindezek olyan feladatok, amelyeket algoritmusok felhasználásával végzünk el. Ezeket az algoritmusokat pedig pontosan ilyen feladatok megoldására fejlesztették ki. Így ezeket a speciális matematikai feladatokat igen hatékonyan lehet megoldani.

Egy algoritmus tehát semmi más, mint az a számítási előírás, amellyel a számítógép tudomására hozzuk, hogyan oldja meg — azaz hogyan számolja ki — az adott feladatot.

Egér

Az egér egy kb. cigarettadoboz méretű beviteli eszköz, amellyel a kurzort igen gyorsan és kényelmesen lehet a képernyőn pozícionálni. Főleg a grafikus programokat lehet az egérrel sokkal komfortosabban és pontosabban kezelni, mint a joystickkel. Viszont az egér használata feltételezi egy olyan program meglétét is, amely az egér által küldött jeleket megfelelően földolgozza. Mialatt a kezünkkel mozgatjuk az egeret az asztalon, ennek megfelelően változik a kurzor helyzete is a képernyőn. Az egéren lévő billentyűkkel a használt programtól függően parancsokat adhatunk be.

Két konstrukciós elvet különböztethetünk meg az egerek esetében. Az egyik szerint a mozgást szenzorokon keresztül egy golyó segítségével adjuk át (mechanikus egér), a másik fajta megoldásban foto-

cellákat alkalmazunk. A fotocella akkor reagál, ha az egeret egy speciális felület fölött mozgatjuk (optikai egér). Az egér a nevét onnan kapta, hogy annak kialakítása („szemgombocskák”, kábelfarkinca) igen hasonlít egy igazi egérhez.

Grafikus tábla (digitalizáló tábla)

Ez egy olyan berendezés, amely lehetővé teszi a papíron lévő rajzok gépbe való vitelét. Ehhez a papírt a lapra kell fektetni, majd egy speciális ceruzával végig kell menni a rajzon. A ceruza pozícióját a forgatható és hajlítható karon keresztül érzékeljük, vagy pedig a tábla speciális raszterfelülete érzékeny. A megadott pontokat (rajzot) digitális formában adjuk a számítógépnek, ahol azután azokat tovább fel lehet dolgozni. Egy ilyen rendszer természetesen a szabad kézzel elkészített rajzokat is „elfogadja”! Jelenleg a házi számítógépekhez ilyen grafikus táblát (digitalizálót) nem készítenek.

Fényceruza (Lightpen)

Egy fényceruza csakis egy katódsugaras képernyővel együtt működik. Általában a kapható készülékeknél ez a szokásos megoldás. A fényceruzával a felhasználó érintéssel rögzíthet pontokat a képernyőn. A számítógép a ceruzába épített fotocella segítségével határozza meg a katódsugár időbeli változásait, ezzel pedig meghatározza a pillanatnyi helyzetet is. Ezt az egyszerűen kezelhető eszközt például grafikus menük kiválasztására lehet használni, vagy mondjuk a képernyőre történő rajzolásra. Korábban a fényceruzát csupán a nagygépes CAD rendszereknél lelhettük föl, ma viszont ezt a készüléket a házi számítógépek számára is kínálják megfelelő szoftverekkel. Meg kell azonban jegyezni, hogy az alkalmazott eljárás pontatlanságai miatt a fényceruzák egzakt pozícionálásra nem lehet használni.

Reset védelem



Ha szeretnénk egy BASIC programot resetállóvá tenni, használjuk listánkat. Ez lekapcsolja a RUN/STOP RESTORE billentyűket, plusz a LIST rutint is. A lényeg, hogy a BASIC program rendelkezzen egy nullás sorral, mivel a reset helyett RUN 0 utasítást hajtunk végre. Ha ezt az újraindítást szeretnénk a RUN/STOP RESTORE megnyomása után is elérni, akkor a programban legyen benne a POKE792,226:POKE793,252 utasítás. A programunkat ,8,1-gyel hívjuk be és a SYS 49152-vel indítsuk. A reset védelem ettől kezdve a gép kikapcsolásáig aktív.

```
1 REM ********************************
2 REM *                              *
3 REM *       RESETVEDELEM           *
4 REM *                              *
5 REM ********************************
6 :
7 :
8 PRINT CHR$(147)"DATA BEOLVASAS ES ELLENORZES ...";:J=49152:VE=49231:P=J
9 FOR B=0 TO 7:READ A$
10 L=ASC(MID$(A$,2,1))
11 H=ASC(MID$(A$,1,1))
12 L=L-48:IF L>9 THEN L=L-7
13 H=H-48:IF H>9 THEN H=H-7
14 PRINT"■■■■■■■■■■■■"P;:P=P+1
15 IF H>15 OR L>15 THEN 17
16 A=H*16+L:POKE J+B,A:T=T+A:NEXT B:READ A:IF A=T THEN 18
17 PRINT:PRINT"DATA HIBA ...  SOR:"PEEK(64)*256+PEEK(63):END
18 T=0:J=J+8:IF J<VE THEN 9
19 PRINT"■■KESZ":END
20 DATA A9,31,A2,C0,8D,00,80,8E,0993
21 DATA 01,80,8E,03,80,A9,45,8D,0791
22 DATA 02,80,A9,C3,A2,C2,A0,CD,1215
23 DATA 8D,04,80,8E,05,80,8C,06,0694
24 DATA 80,A9,38,A2,30,8D,07,80,0839
25 DATA 8E,08,80,A9,EA,8D,28,03,0965
26 DATA 60,A9,00,8D,03,08,8D,04,0562
27 DATA 08,20,A3,FD,20,8E,A6,20,0828
28 DATA 5E,A6,4C,71,A8,58,A8,68,0993
29 DATA AA,68,40,AA,68,40,2E,FF,0977

READY.
```

Még többet ésszel!

V. rész

Remélem, kellemes fejtörést okoztam a múlt havi feladványokkal. Mindenki megnyugtatására hadd szögezzem le újra: az általam kínált megoldás nem szükségképpen az egyetlen vagy a legjobb változat, de remélhetőleg hibátlan, és egyike a legésszerűbbeknek. És sosem lemásolásra, hanem elemzésre ajánlom.

Az első feladat (keresés parancssorban) megoldása az 1. példa. A programsor nem tartozik hozzá, csak azért készült, hogy legyen miben keresni.

Lehet, hogy nem mindenki érti a függvény második paraméterét. Az egyenlőségjel egy utasítás elején, egy változó után értékadást jelent; minden egyéb helyen viszont ún. relációs (hasonlító) logikai művelet jele. Jól megfigyelhető ez a kettős szerep az előző rész 1. példájának 290-es sorában, ahol még a zárójellel sem lett volna kötelező a relációt megkülönböztetni. Ez egy nagyon hasznos és egyszerű eszköz, ezért hadd foglaljam össze a lényegét.

Relációt három jellel vagy azok kombinációival írhatunk elő, ezek a következők: '⟨, '=', '⟩'. A művelet két azonos típusú kifejezés között végezhető el, tehát karakterláncokkal is. Ilyenkor a karakterek ASC-kódjának sorrendje dönt, azzal a kiegészítéssel, hogy az üres karakter a nullásnál is kisebb, vagyis az ,,AB"⟨,AB!" művelet eredménye igaz.

Műveletről, eredményről beszélek, de hát mit is jelent az ,,igaz" a gép számára? Ősrégi megállapodás szerint az 1 (magas) bitet ,,igaz", a 0 bitet ,,hamis" értékként emlegetik. Mivel a BASIC az egész számokat két byte-on ábrázolja, ezért az ,,igaz" a BASIC-ben tizenhat (2 byte) egyes bit, a ,,hamis" tizenhat nullás alakjában fejezhető ki. A BASIC nyelvek általában csak a tizes számrendszert ismerik, ezért a két logikai értéket is így kell kifejeznünk. Az ,,igaz"-nak így —1, a ,,hamisnak" 0 lesz a számszerű megfeleltetése. Azt, hogy az egyesek 65355 helyett miért —1-et jelentenek, majd a többi logikai művelet kapcsán fogom elmesélni.

A relációs műveletek értékei tehát számok, amelyek egyszerűen átkapcsolhatók egymásba (2. pl), és amelyekkel matematikai műveletek végezhetők, például szorzás. Ennek során vagy nulla értéket kapunk, vagy az eredeti érték negáltját. Ezt a tulajdonságot használtam ki a RIGHT$ függvény (1. pl.) második paraméterének megválasztásakor, a visszakapott karakterlánc hossza 0 vagy 4 lesz.

Másik remek tulajdonsága a relációnak, hogy értékei egymást követő számok. Ez azért is jó, mert alkalmas az ON utasítás kapcsolójának feladatára.

Az ON olyan utasítás, ahol több címhivatkozás közül egy numerikus kifejezés értékével jelölhetjük ki az éppen végrehajtandót. Érthetőbb lesz,ha szemügyre vesszük a 3. példát, és annak ON-os behelyettesítését (4. pl). A V helyére persze bármilyen kifejezés, tehát logikai műveleteket tartalmazó kifejezés is beírható. Figyelemre méltó adat, hogy az ON ebben a helyzetben határozottan gyorsabb az IF-nél, amire — már mondtam néhányszor — egy idő után elkezd odafigyelni a programozó ember.

Kis kitérőként nézzük már meg, hogy milyen alapon állítom én egy utasításról, hogy gyorsabb egy másiknál. A mikroszámítógépekben rendszerint található egy belső óra, amely állása lekérdezhető. A C64-ben, amelyen én dolgozom, ez a TI és TI$ nevű változók révén oldható meg. A TI értékét a megszakító-rutin másodpercenként hatvanszor eggyel növeli, vagyis ez az óra legnagyobb pontossága. A BASIC utasítások végrehajtása, még ha ez lassúnak is számít, 1/60 másopercnél azért sokkal kevesebb ideig tart. Ha tehát pontosan akarjuk ennek időtartamát megállapítani, akkor fel kell nagyítanunk a vizsgálat tárgyát, mondjuk, ezerszeresére. Magyarul ezerszer kell végrehajtani a vizsgált utasítást egy ciklusban. Bár ez nem teszi lehetővé egy végrehajtás valódi idejének megmérését, alkalmas az utasítások összehasonlítására. A minta (7.pl) 6. sorába lehet a vizsgálandó utasítást beírni, de természetesen ez a rutin a szükség szerint átvariálható.

Ott tartottunk, hogy mit csinál az ON. Nem szükséges, hogy a kapcsolónak használt érték 1 és valami közé essen, hiszen ez egy összeadással (és esetleg egy szorzással) bármikor létrehozható (5. pl). Itt meg is figyelhető egy reláció alkalmazása az ON-ban. Szubrutin-hívások is kapcsolhatók (6. pl), ekkor a RETURN hatására az ON után folytatódik a program. Ha a kapcsoló értékkészlete nem folytonos, akkor a nem használt címkéket ki is hagyhatjuk. Két dologra kell csak ügyelni ha a kapcsoló felvehet egy értéket, akkor a címlistában legyen neki megfelelő sorszám; és a kapcsoló sose legyen negatív. Ha a kapcsoló nulla, vagy nagyobb, mint ahány sorszámot felsoroltunk, akkor nincs címhívás, hanem az ON utáni utasításon (itt a STOP) folytatódik a futás.

Nem árultam még el az előző rész másik feladványának megoldásait. Négy kis ujjgyakorlatot ígértem a pozícionálásra, egy apró ciklusba (8. pl) ágyazva. A megoldások a 9. példán láthatók. Nem sok magyarázkodást igényelnek, de azért mindhez van egy megjegyzésem:

(1) Azért elég hat szóköz, mert a STR$ legalább két karakter hosszú. (2) Azért 21, mert a hosszba az előjel is beleszámít. (3) Egy szám tízes alapú logaritmusának egész része eggyel kevesebb, mint a szám (egész részének) hossza. Például lg438=2,64. A tízes alapú logaritmus idegen (itt ún. természetes) alapból az itt látható osztással számítható ki. (A matekban is vannak jó trükkök.) Kár, hogy nullára nem működik. A gép számítási pontossága megfelel a feladat igényeinek. (4) A MID$ a szóköz lecsípésére kell.

És most egy egészen más téma következik. Ezeket a sorokat júniusban írom, így a legutolsó megjelent lapszám a 92/6-os. Ebben a 14. oldalon bemutatok egy kis programrészletet, amely kijavítása és egyszerűsítése volt a házi feladat. Nos, két olvasónak annyira felcsigázta az érdeklődését a probléma, hogy a szerkesztőségen kereresztül levélben elküldték megjegyzéseiket. Ezek szolgáltak néhány tanulsággal, amelyeket most közzé szeretnék tenni.

Pusztafi Gábor adott egy jó megoldást, de ezenfelül levélében kifogásolta, hogy a program sorai nincsenek egyenletesen átszámozva, hogy értelmetlennek látszik az RND értékkészlete, hibásnak ítélte a WAIT utasítást, és ennek kapcsán szükségesnek vélte a Stop billentyű letiltását.

Örömömre szolgál, hogy van más is, aki hozzám hasonló elveket vall a program sorszámairól, itt azonban a megjelent változatot védem. Egyrészt a 0. és 99. sorok nem az eredeti programból származnak, hanem én írtam őket azért, hogy a kiemelt rész futtatható legyen. A távoli sorszámokkal a ciklusmag önállóságát akartam jelezni. Másrészt — és ez a mondandóm lényege — a bemutatott példákat nem kell kész programnak tekinteni,

és elvárni tőlük egy kész program összes vonását. A példákat képzelhetjük egy nagyobb program belsejébe, és mindig feltételezhetjük, hogy a hiányzó alapfeltételek valahol teljesülnek. Nem is kell ilyen dolgokkal foglalkoznunk, hanem teljes figyelmünket összpontosíthatjuk a példában bemutatott fogásra, ötletre, tanulságra. Vonatkozik mindez a Stop letiltására is, amely nagyon lényeges egy kész programban, de felesleges egy rövid életű példában; és az RND-re is, amely intervallumáról most már nem is érdemes beszélni.

A WAIT nem volt hibás, legfeljebb nem tökéletes. Ugyanis a program itt addig vár, amíg egy billentyűt le nem ütök — a DEL kivételével. A gyengéje ellenére hasznos lehet ez az ötlet, és nagyon rövid is.

Legyen tehát mindenki nyugodt: nem vedelem a kólát, miközben vizet prédikálok. A komolyabb programjaimban én is betartom a magam alkotta szabályokat, csak így figyelhettem meg a hasznosságukat.

A másik levél, Mesterházi Sándortól, szűkszavúbb volt: a feladat négy, fokozatosan fejlesztett megoldását tartalmazta. Érdekes dolog, hogy más úton jutott ugyanehhez a megoldáshoz, mint amelyet én akkoriban bemutattam.

Nem szabad azonban elmennem amellett, hogy mindhármunk megoldása különbözik a csillag kiírásának oszloppozíciójában. Lehetséges, hogy a feladatból mindenki számára másként derült ki az eredeti szerző szándéka, ezért óvakodok attól, hogy az enyémtől eltérő megoldások készítőit hibáztassam.

Szerintem nagyon fontos dolog, hogy ha egy feladatot meg kell oldanom, főleg ha egy program részletét kell kijavítanom, akkor szigorúan ragaszkodjak az eredeti feladat által meghatározott kimeneti formátum betartásához. Nem tudhatom ugyanis, hogy az általam megváltoztatott tulajdonságokat a teljes program miként használta ki, és mit idézek elő a változtatással. Ha a teljes programot nincs időm elemezni, vagy mások által párhuzamosan készített részekhez kell illeszkednem, akkor kockázatos, azaz helytelen dolog az ellenőrizetlen javítgatás. Még ha csak ilyen apróságról van is szó, mint egy TAB pozíció.

Persze nem feledtem, hogy az előbb a példák részleteinek jelentéktelenségét ecseteltem, de itt most egy elvről van szó, amely figyelmen kívül hagyása sok kínos percet okozhat.

Következik az új házi feladat. Meg kell mondanom, hogy a feladványok megoldásának beküldéséért semmi jutalmat nem adhatok, még a megjelenésüket sem ígérhetem, de érdeklődéssel fogadom őket, még inkább a megjegyzéseket és javaslatokat.

A 10. példa egy véletlenszerű értékekkel feltöltött vektorban való maximumkeresést végez el. Kiírásra kerül a legnagyobb elem (azonosak közül az első) indexe, és a maximum. Az előző feladathoz hasonló okokból szükségünk támadt a 3–7. sorok helyettesítésére egyetlen parancssorral. Két különböző megoldást fogok bemutatni. Véleményem szerint ez a csemege bizonyíték a BASIC rugalmasságára és sokoldalúságára.

Hódi Gyula

```
(1.)
1 DIM N$(500): FOR I=1 TO 1000: N$(RND(1)*500+1)="*": NEXT I
FOR I=1 TO 500: PRINT RIGHT$("    "+STR$(I),-4*(N$(I)=""));: NEXT I

(2.)
1 F=-1-F: PRINT F: GOTO 1

(3.)
10 IF V=1 THEN 70
20 IF V=2 THEN 90
30 IF V=3 THEN 140

(4.)
10 ON V GOTO 70,90,140

(5.)
10 ON (V$="I")+2 GOTO 50,30

(6.)
10 ON K GOSUB 100,,80,: STOP

(9.)
2 PRINT RIGHT$("        "+STR$(C),8)
2 PRINT TAB(21-LEN(STR$(C)));C
2 PRINT TAB(19-INT(LOG(C)/LOG(10)));C
2 PRINT ,RIGHT$("0000"+MID$(STR$(C),2),5)

(7.)
1 TI$="000000"
5 FOR I=1 TO 1000
6 :
9 NEXT I: PRINT TI/60

(8.)
C=INT(RND(1)*20000)+1
2 PRINT C
3 NEXT I

(10.)
1 DIM N(500)
2 FOR I=1 TO 500:
  N(I)=INT(RND(1)*1000): NEXT I
3 X=0: N(X)=-1
4 FOR I=1 TO 500
5 IF N(I)>N(X) THEN X=I
6 NEXT I
7 PRINT X,N(X)
```

Képernyő varázslat – a TED

Első rész: A hardver

Ez egy új cikksorozat első száma, mellyel azok életét szeretném megkönnyíteni, akik — mint magam is — a COMMODORE 16/116/PLUS4 gépek valamelyikét programozzák, és már megtanulták a programozás alapjait. Az ő számukra igen fontos, hogy tudásukat valamire fel is használják, kipróbálják a gyakorlatban, hogy mit is tud számítógépük. Bár a sorozatban leközölt programok gépi kódban készültek, egy részük a BASIC programozók számára is hasznos lesz, egy más részük sajnos sebességi okok miatt csak gépi kódban fog működni. Át fogjuk tanulmányozni a TED-hez kapcsolódó összes funkciót (grafika, hang, megszakítások, billentyűzetkezelés, memóriák), példaprogramokkal illusztrálva, melyek — ha nem emelem ki a különbséget — JCL assebmler-listában kerülnek közlésre. Itt szeretném megjegyezni, hogy ha a továbbiakban PLUS/4-et írok, a gépcsalád összes tagját értem ez alatt, hacsak nem emelem ki az esetleges különbözőséget. Néha a könnyebb érthetőség kedvéért szólni fogok az elektronikai oldalról is, bár tartok tőle, hogy sokan nem értenek egyet azzal, hogy az elektronika bármit is érthetőbbé tehet. Nem fogok elveszni ennek részleteiben, de sok dolgot csak ennek segítségével lehet megérteni. A rövidítéseket részletesen meg fogom magyarázni, először az alkotó szavak megadásával, a rövidítésben előforduló betűket nagybetűvel írva, majd a jelentés és az esetleges kiegészítés leírásával.

A hardver

El is kezdeném fenyegetésem beváltását. Kérek mindenkit, hogy olvassa el ezt a részt, a későbbiekben fogok még rá hivatkozni. Mi is hát a hardware (magyaros helyesírással ,,hardver")? Az angol ,,hard" szó keményet jelent, a ,,ware" pedig árut. A ,,hardware store" pedig a magyar ,,VASEDÉNYhez" hasonlít. Ezek után persze ne fazekakra gondoljunk a ,,hardware" szó hallatán, hanem valami olyanra, ami kemény, és nem tudjuk megváltoztatni, például a gép kapcsolása, az alkalmazott áramkörök, a NYÁK (Nyomtatott Áramkör) stb. (és a többi).

A TED

A TED — eme név egy betűszó, az angol TExt Display rövidítése, ami magyarul szöveges megjelenítőt jelent. Igencsak alábecsülnénk azonban a TED képességeit, ha elhinnénk, hogy ez a nagy integráltságú céláramkör csak ennyire képes. Hogy képet alkothassunk valódi tudásáról, vessünk egy pillantást a gép kapcsolási rajzára!

Keressük meg a TED-et! Típusszáma 7360 vagy 8360 (a német kézikönyvben az első számmal szerepel). ,,°" a PLUS/4-es gép egyik legfontosabb áramköre. A gépben ezen a helyen egy 48 lábú DIL tokot találunk (DIL=Dual-In-Line, a tok két oldalán vannak a lábak), ebben helyezkedik el a nagy varázsló.

Látszólag e 48 láb közül mindössze három (illetve szorosan véve csak kettő) foglalkozik az IC (Integrated Circuit = integrált áramkör) nevében foglaltakkal, azaz a kép- (és hang-) előállítással, a 13-as Colour (szín), a 23-as Lum (fényerő) (és a 33-as SND (hang)) láb, melyek a modulátorra csatlakoznak. Mint azt még látni fogjuk, a látszat csal: a TED legtöbb lábának van köze a megjelenő képhez.

A maradék 45 láb közül kettő az áramkört a tápvezetékkel köti össze (4-es láb: +5 V, 24-es láb: GND-föld).

A TED rendelkezik saját cím- (A0—A15-ös bitek: 1—3-as és 36—48-as lábak), valamint adatbusszal (D0—D7-es bitek: 25—32-es lábak), melyekkel a teljes memóriát látja (azért csak ,látja", mert a memóriát csak olvasni képes, írni nem). Azért kell olvasnia a memóriát, hogy adatokat ,,hozhasson el" a képelőállításhoz (például a karakterek képét a karaktergenerátorból). Nyolc vezetékkel a billentyűzetre és a botkormányokra csatlakozik (KO—K7-es bitek: 15—22-es lábak).

A MUX (MUltipleX) jel (9-es láb) a DRAM-ok (dinamikus RAM-ok (RAM — Random-Acces Memory — közvetlen hozzáférésű memória, ami azt jelenti, hogy a memória bármely rekeszét azonos idő alatt érjük el), olyan RAM-ok, melyek belül apró kondenzátorokból épülnek fel) címmultiplexelését vezérli a RAS (Row Address Strobe — sorcím kiválasztás, 10-es láb) és CAS (Column Address Strobe — oszlop cím kiválasztás, 11-es láb) jelekkel együtt. A DRAM-ok ugyanis a 16-bites címinformációt nem közvetlenül olvassák be a címbuszról, hanem 2*8 bitben. Hogy 2*8 mióta nem egyenlő 16-tal? Nos, valójában egyenlő, de mégsem. Most a magyarázatot a PLUS/4-ben alkalmazott (Figyelem! CSAK a PLUS/4-ben!) 4164 típusjelű RAM leírásával folytatom, de a leírtak hasonlóan játszódnak le a többi DRAM-ban is. A 4164-es DRAM-ok kapacitása tokonként 64kbit (64*1024=65536 bit). Ez a jókora mennyiség egy 256*256-os méretű belső mátrixban helyezkedik el, mely sorokból és oszlopokból áll. Tehát: egy DRAM olvasásakor a cím kikerül a közös 16 bites címbuszra (közös, mert az egész gépé, kivéve a DRAM-okat). Ott rákerül a multiplexeráramkörök bemenetére. A TED a MUX jelével — mely a 74LS257 típusú multiplexer SEL A (adatválasztó) bemenetére kerül — jelzi, hogy a címinformáció felső 8 bitjét küldje ki a multiplexer a DRAM-ok külön, 8bit széles címbuszára. Ezzel egyidejűleg aktiválja a DRAM-okba befutó RAS jelet, jelezve, hogy megérkezik a sorkiválasztáshoz szükséges információ. Ezt a RAM beolvassa, az adott sorszámú sort átmásolja egy belső pufferbe. Mivel azonban ilyenkor az adott sor kis kondenzátorai kisülnek, a pufferből rögtön vissza is írja a sort,azaz a kondenzátorokat újból feltölti. Kis idő múlva a TED a MUX jel állapotát megváltoztatja, és ,,kimegy" az alsó nyolc bit, azaz az oszlop száma. Törli a RAS, aktiválja a CAS jelet. A RAM ezt érzékelve beolvassa az oszlopszámot, és a pufferből az adott sorszámú bitet az adatbuszra teszi. Mivel minden RAM kitesz egy-egy bitet az adatbuszra, ott egy bájt jelenik meg. A MUX jel újra vált. Mivel befut a processzor GATE IN (23-as láb) bemenetére, amit az végig figyelt, a processzor tudja, hogy lejátszódott egy teljes ciklus, az adatbuszon ott az adat,amit most elhozhat. Most már talán látszik, hogy mi a különbség a 16 és a 2*8 bit között. Hasonló módon történik

a memóriafrissítés is. Ilyenkor a processzor megállítása után (erről is mindjárt lesz szó) a TED minden sort megcímez (azaz a RAS jelet kiküldi), mire az adott sor kiolvasódik és újra beíródik — azaz felfrissül, de CAS jelet nem küld, így semmi egyéb nem történik. A memóriafrissítésre a RAM-ok dinamikus volta miatt van szükség, enélkül a tárolt adatok elvesznének (a kis kondenzátorok előbb-utóbb maguktól is kisülnek). A DRAM-ok használatát alacsonyabb áruk indokolja, valamint az, hogy egyszerűbb szerkezetük miatt a statikus (frissítést nem igénylő) RAM-oknál nagyobb kapacitást lehet belezsúfolni egy ugyanakkora tokba.

Még egy, a RAM-okkal kapcsolatos jelről kell néhány szót ejtenem. Ez a jel a TED R/W bemenetére érkezik, és általában a processzor vezérli (Read/Write — olvasás/írás). Használata nyilvánvaló: a RAM-ok írását/olvasását vezérli azok WE (Write Enable — írás engedélyezés) bemenetén (TED R/W : 7-es láb, 4164 RAM chipek : 3-as láb). Természetesen 16k-s gépben más RAM-ok vannak: C—16, C—116-os chipekből van két darab. Ezek címtartománya 16kbit, de 4 bit „szélesek". Ez azt jelenti, hogy négy lábukkal kapcsolódnak az adatbuszra, így olvasáskor oda négy bitet küldenek tokonként, míg íráskor négyet fogadnak. Bővítéskor ezeket cserélik le 4464-esekre — 64 kbit címtartomány, 4 bit széles, természetesen szintén két darabra — vagy 8 darab 4164-esre — 64 kbit címtartomány egy bit szélesek, úgy, mint a PLUS/4-ben. Természetesen mindezen RAM chipek rendelkeznek WE lábbal. Mivel a RAM-ok WE jele logikai nulla szinten aktív, az „1" a RAM olvasását, míg a „0" annak írását jelenti.

Jogosan merül fel a kérdés: miért nem „akad össze" a processzor és a TED, hiszen memóriakezeléskor zavarhatják egymást? Nos, a válasz ismét egyszerű: a TED elintézi ezt is, oly nemes egyszerűséggel, hogy öröm nézni. Egyszerűen „lekapcsolja" a processzort a buszról, ha neki van rá szüksége. Erre szolgálnak az AEC (Address Enable Control — cím engedélyezés vezérlés) és BA (Bus Available — busz elérhető) jelek. A nevektől (különösen az AEC első olvasásra teljesen értelmetlen nevétől) nem kell megijedni. Az AEC azt jelzi a processzornak, hogy a buszt használhatja-e vagy sem, a BA jellel a processzor működése függeszthető fel arra az időre, amíg valami más (alapesetben a TED, de bármely más — általunk csatlakoztatott — hardware eszköz) a tárat használja. Ez leggyakrabban akkor fordul elő, mikor a TED a memóriát frissíti, illetve mikor adatokat hoz el a képelőállításhoz. Minden karaktersor előtt elhozza a képernyőmemóriából a karakterkódokat, a színmemóriából a színkódokat, és minden egyes sor elején a karaktergenerátorból a sor bittérképéhez szükséges adatokat. Természetesen a bittérképes (bittérkép vagy angolosan bitmap — az a tárterület, megy a képernyőt pontokként ábrázolja, például a BASIC GRAPHIC utasítással érhető el) üzemmódban ez másképp történik, ilyenkor a karaktersor elején a szín- és fényerőinformációkat hozza el, míg a rasztersorok elején a sorok bittérképét. A TED ilyenkor — vagyis az alsó és felső keret „ábrázolásának" kivételével mindig — a processzor „phi null" órajelét felezi. Ezt a felezett órajelet „SINGLE CLOCK"-nak hívjuk (jelentése : egyszeres órajel), frekvenciája 886,7 kHz, az alsó és felső kereten pedig 1,773 MHz frekvenciájú (egyébként ez a maximális frekvencia) órajelet kap a processzor. Ennek neve „TWICE CLOCK", azaz kétszeres órajel. Erről még lesz szó a programozásnál is. Az „phi null"-t a processzornak a 12-es lábán küldi el a TED, melyet az az 1-es lábán fogad. A TED az órajelet a CLKIN (órajel be : 14-es láb) bemeneten kapja, és ezt osztva állítja elő a „phi null"-t. Az IRQ (Interrupt ReQuest — megszakítás kérelem) jel a TED 8-as lábáról fut be a processzor 3-as lábára, és alacsony (logikai nulla) szinten aktív. A processzorba érkezve a megszakításkérelmet a processzor vagy figyelembe veszi, vagy sem, ez az „I" (Interrupt — megszakítás) bittől függ, melyet a programozó állíthat be a processzor állapotregiszterében (természetesen csak gépi kódban). A megszakításokról a későbbiekben részletesen lesz szó. Már csak két jel maradt: a CS0 és a CS1 (Chip Select — chip kiválasztás) az 5-ös és 6-os lábakon. A CS0 azt jelzi, hogy a ROM-ok (Read-Only-Memory — csak olvasható memória) közül az alacsony („Low", $8000-$BFFF) vagy a magas („High", $C000-$FFFF) terület legyen-e érvényes. Erre azért van szükség, mert a PLUS/4-ben használt ROM IC-k kapacitása 16kByte, így címbuszuk is csak 14 bit széles, és mivel ezzel a 14 bittel a címbusz alsó bitjeit figyelik, nem tudnák, mikor van róluk szó, ha mindig aktívak lennének. Ezt az aktivitást vezérli a két Chip Select jel, melyeket úgy is elfogadhatunk, mint egy plusz címbitet. A PLUS/4-ben 16k BASIC, 16k KERNAL, 16k FUNCTION Low és 16k FUNCTION High ROM van (ezek közül a FUNCTION ROM-ok csak a PLUS/4-ben találhatók meg, ugyanis ezek tartalmazzák a beépített programokat). A ROM-okról a későbbiekben még lesz szó.

Remélem ez a rövid és meglehetősen tömör leírás senkit sem kedvetlenített el, és most már senki sem néz olyan csúnyán a gép kapcsolási rajzára. Minden fellelkesült újdonsült „hardware-szakembert" szeretnék itt figyelmeztetni, hogy tapasztalt ismerős jelenléte nélkül inkább ne próbálkozzék semmilyen munkával (például sk. (Saját Kezű) memóriabővítés), mert ennek a gép 99% eséllyel csak kárát láthatja.

Viszlát a következő számban (mármint a COMMODORE újságában)!

Mentőöv C128-as szövegekhez

Előfordul néha, hogy egy szövegszerkesztő lemerevedik. Ez persze rossz, de különösen akkor, ha egy hosszabb szöveget írtunk, de letárolni elfelejtettük. Most nehogy újra nekikezdjünk a munkának. Előbb próbáljuk ki a következőket: (A most leírt példa a *Mastertext 128*-ra vonatkozik!)

Nyomjuk meg a reset gombot, hogy kirántsuk a gépet a letargiából. Hívjuk fel az⟨F8⟩ gombbal a monitort. Közvetlenül is beléphetünk a monitorba a reset+⟨RUN/STOP⟩ gombbal, de ilyenkor gondok lehetnek a tárolókezeléssel.

Keressük meg most a **H 10400 1FF00 00** utasítással a szövegünk végét. A C128 kiadja azt a címet, ahol a lezáró jelet megtalálta. Tároljuk le tehát az adott tartományt a következő utasítással: **S "szövegnév...T",8,10400,végcím**

A „végcím" helyére természetesen a megtalált számot kell írni. A szövegnév 15 karakteres legyen, a 16. pozíció a nagy T helye. Ezt se felejtsük el. A floppyra mentett szöveg máris újra behívható.

Ez a trükk természetesen minden egyéb C128-as szövegszerkesztővel használható. Csak a szöveg kezdete-vége címeket kell előkeresni. Ha ezt nem tudjuk, hívjunk be egy hosszabb szöveget, az elejére és a végére írjunk könnyen megjegyezhető, speciális karaktert vagy karaktereket (pl, ABCD, @, stb.), reseteljük a gépet, majd keressük meg a monitorral az adott kódokat. Ebből megtudhatjuk, hol kezdi el a szöveget az adott program, és milyen kódot tesz a szöveg végére.

Directory laponként

Ha sok bejegyzés van egy lemez tartalomjegyzékében, akkor a DIRECTORY parancs kiadásakor az elől állók egyszerűen kigördülnek a képernyőről, ami boszantó. Ráadásul nem mindenki olyan villámkezű, hogy időben elérje a ⟨NO SCROLL⟩ gombot. A Directory 25 Z program megoldja a problémát. A program használatával a directoryt három oszlopban írjuk a képernyőre.

A programot a BANK 0:BLOAD "DIRECTORY 25.Z." utasítással töltsük be, és a SYS 4864-gyel inicializáljuk. Ha ezután használjuk a DIRECTORY vagy a CATALOG parancsokat, a tartalomjegyzék három oszlopban jelenik meg a képernyőn. Ha a kijelzés befejeződött, netán a 80 karakteres képernyőt teleírnánk, a rutin addig vár, amíg meg nem nyomunk egy billentyűt.

C128-as tippek, trükkök

```
10 REM ****************************
20 REM *                          *
30 REM * DIRECTORY LAPONKENT      *
40 REM *                          *
50 REM *          C128            *
60 REM *                          *
70 REM ****************************
80 REM
90 FOR I=4864 TO 5689
100 READ A:POKE I,A:C=C+A
110 NEXT I
120 IF C<>79661 THEN PRINT "HIBA A DATA-SOROKBAN...":END
130 END
140 DATA169,31,162,19,141,12,3,142,13,3,169,55,162,19,141,14,3,142,15,3,169
150 DATA84,162,19,141,16,3,142,17,3,96,170,169,19,160,111,32,226,67,16,9,41
160 DATA127,24,105,39,162,0,240,2,56,138,76,33,67,224,255,240,21,201,39,144
170 DATA17,201,45,176,13,56,233,39,9,128,170,169,19,160,111,76,106,81,56,76
180 DATA205,81,201,39,144,19,201,45,176,15,56,233,39,10,170,189,131,19,72,189
190 DATA130,19,72,24,36,56,76,169,75,46,68,73,82,69,67,84,79,82,217,46,67,65
200 DATA84,65,76,79,199,0,133,19,133,19,169,0,141,0,255,32,66,193,169,0,133
210 DATA144,169,8,32,177,255,169,240,32,147,255,169,36,32,168,255,32,174,255
220 DATA169,8,32,180,255,169,96,32,150,255,169,0,32,169,21,16,3,76,179,21,32
230 DATA193,19,32,115,21,76,220,19,169,1,141,9,11,141,12,11,169,0,141,8,11
240 DATA141,7,11,169,4,141,10,11,169,21,141,11,11,96,160,0,32,169,21,144,3
250 DATA76,230,21,200,192,7,208,243,152,72,169,0,141,7,11,162,24,160,1,32,245
260 DATA20,169,18,32,210,255,104,168,32,169,21,144,3,76,230,21,200,32,210,255
270 DATA192,23,208,240,32,125,255,32,32,32,0,32,169,21,144,3,76,230,21,201
280 DATA34,208,3,76,23,20,32,210,255,200,192,30,208,233,169,146,32,210,255
290 DATA32,250,20,32,169,21,144,3,76,230,21,201,1,240,3,76,54,20,32,169,21
300 DATA144,3,76,230,21,32,169,21,144,3,76,230,21,133,250,32,169,21,144,3,76
310 DATA230,21,166,250,32,50,142,32,8,21,169,34,32,210,255,32,169,21,144,3
320 DATA76,230,21,201,32,240,244,201,34,240,3,76,201,20,32,169,21,144,3,76
330 DATA230,21,201,34,240,6,32,210,255,76,127,20,32,19,21,169,34,32,210,255
340 DATA32,169,21,144,3,76,230,21,201,35,176,3,76,153,20,201,32,240,14,32,210
350 DATA255,32,169,21,144,3,76,230,21,76,168,20,169,13,32,210,255,238,7,11
360 DATA32,30,21,76,51,20,36,32,125,255,27,27,20,32,18,66,76,79,69,67,75,69
370 DATA32,70,82,69,73,146,0,162,23,160,29,32,32,22,32,245,20,32,230,21,32
380 DATA165,21,76,55,22,32,66,193,24,32,240,255,96,238,8,11,174,8,11,172,9
390 DATA11,24,32,240,255,96,174,8,11,172,10,11,24,32,240,255,96,174,8,11,172
400 DATA11,11,24,32,240,255,96,173,7,11,201,22,208,53,238,12,11,173,12,11,201
410 DATA6,208,6,32,21,22,76,91,21,238,12,11,173,9,11,105,26,141,9,11,173,10
420 DATA11,105,26,141,10,11,173,11,11,105,26,141,11,11,169,0,141,7,11,169,0
430 DATA141,8,11,96,162,24,160,29,24,32,240,255,32,141,21,165,213,201,88,240
440 DATA250,32,193,19,32,115,21,96,24,169,0,162,0,32,45,192,56,169,23,162,79
450 DATA32,45,192,32,66,193,32,125,255,19,19,0,96,32,125,255,18,32,84,65,83
460 DATA84,69,32,68,82,85,69,67,75,69,78,32,146,145,0,96,32,171,255,96,32,165
470 DATA255,56,166,144,208,1,24,96,16,49,162,23,160,24,32,245,20,32,125,255
480 DATA70,76,79,80,80,89,32,78,73,67,72,84,32,69,73,78,71,69,83,67,72,65,76
490 DATA84,69,84,0,32,21,22,32,165,21,32,91,21,76,55,22,162,23,160,29,32,245
500 DATA20,32,252,21,32,165,21,32,21,22,32,91,21,76,55,22,169,0,133,144,169
510 DATA8,32,180,255,169,111,32,150,255,32,165,255,32,210,255,36,144,80,246
520 DATA96,160,255,162,255,202,208,253,136,208,250,96,169,8,32,177,255,169
530 DATA111,32,147,255,169,73,32,168,255,169,48,32,168,255,76,174,255,76,66
540 DATA193

READY.
```

PÖTYÖGŐ SZOLGÁLAT

261	+4	91/ 2	80-as képernyő	60.-
262	C64	91/ 2	Fizikateszt	40.-
263	C64	91/ 2	40 Sprite	100.-
264	C64	91/ 2	Raster - Master	60.-
265	C64	91/ 3	Rulett	80.-
266	+4	91/ 3	Tape-disc copy	90.-
267	C64	91/ 3	BASIC rearranger	40.-
268	C64	91/ 3	BASIC decompactor	40.-
269	C64	91/ 3	BASIC merger	40.-
270	C64	91/ 3	SEIKOSHA	60.-
271	C64	91/ 3	Cartridge szimuláció	100.-
272	+4	91/ 3	Sally kulcsszókereső	40.-
273	+4	91/ 3	Merge C+4	60.-
274	C128	91/ 4	1.5 MHz-es C64	80.-
275	C64	91/ 4	Morze - H	60.-
276	C64	91/ 4	Gprint	40.-
277	C64	91/ 4	ASCII - CHR$	40.-
278	C64	91/ 5	Rendező	60.-
279	C64	91/ 5	Keretbeíró	50.-
280	C64 +4	91/ 5	NLQ Print	50.-
281	C64	91/ 5	FLD	40.-
282	C64	91/ 5	Raster Split	40.-
283	+4	91/ 6	Programnyilvántartó	60.-
284	C64	91/ 6	Regiszter	50.-
285	C64	91/ 6	Digitalizált zene	40.-
286	C64	91/ 6	Ékezetes MPS 802	60.-
287	C64	91/ 7	*Ékezetes iratkészítő	80.-
288	+4	91/ 7	Bűvös téglalap	80.-
289	C64	91/ 7	Dominó - H	80.-
290	C64	91/ 7	Raktáros	60.-
291	+4	91/ 7	Tologató	50.-
292	C64	91/ 7	Aren 2000	40.-
293	C64	91/ 7	Bad Lamps	60.-
294	C64	91/ 7	Operation Patriot	40.-
295	+4	91/ 7	Mozaik - B	70.-
296	+4	91/ 7	Logi Kód	50.-
297	C64	92/ 1	Databevivő	40.-
298	+4	91/ 7	Bomber	40.-
299	+4	91/ 7	Dominó - K	40.-
300	+4	91/ 7	The Wall	30.-
301	C64	91/ 7	Mozaik - K	40.-
302	+4	91/ 7	Dominó - R	50.-
303	+4	91/ 7	*Borgok Kincse	150.-
304	+4	91/ 7	Asteroids	50.-
305	C64	91/ 9	Ékezetes irat forrás	30.-
306	C64	91/ 9	Yugo2	40.-
307	C64	91/ 9	Adventure - Sz	100.-
308	C64	91/ 9	Morze Oktató B.	100.-
309	+4	91/ 9	Memory Game Plus	80.-
310	C64	91/ 9	*Capitaly	100.-
311	C64	91/ 9	*Gengszterek	50.-
312	C64	91/ 9	*Küldetés 2000	80.-
313	C64	91/ 9	Itt a $ hol a $	40.-
314	C64	91/ 9	Hewing	40.-
315	+4	91/10	Mikrobi	40.-
316	+4	91/10	Nest of Fleas	150.-
317	+4	91/10	Adventure - F	80.-
318	+4	91/10	A világ szeme	120.-
319	C64	91/10	Gondolkozó	80.-
320	C64	91/10	Memori - B	100.-
321	+4	91/10	Matekastély	80.-
322	C64	91/11	Lottó	60.-
323	C64	91/11	Totó	60.-
324	C64	91/11	Német gyakorló	50.-
325	+4	91/11	Beszélő szótár	100.-
326	+4	91/11	File examiner	50.-
327	C64	91/11	Sprite szerkesztő	40.-
328	C64	91/11	Egyszínű Sprite	40.-
329	C64	91/12	Rulett javítás	40.-
330	C64	91/12	Tőzsde	100.-
331	C64	91/12	IQ teszt	120.-
332	C64	92/ 1	Kalandlap	60.-
333	C64	92/ 1	Levéliró	60.-
334	C64	92/ 1	Képernyőmaszkoló	100.-
335	C64	92/ 1	Sprite Kezelő	80.-
336	C64	92/ 1	SI - OPART	30.-
337	C64	92/ 2	Patience Quartet	100.-
338	C64	92/ 2	Monopoly	100.-
339	C64	92/ 2	Akasztófa	60.-
340	C64	92/ 2	Karaktertervező	50.-
341	C64	92/ 3	String Array Manager	80.-
342	C64	92/ 3	Sprite Basic	40.-
343	C64	92/ 3	Screen Help	40.-
344	C64	92/ 3	Gyors validate	50.-
345	C64	92/ 3	BASIC SYNTAXER I.	150.-
346	C64	92/ 4	Tologató C64-re	80.-
347	C64	92/ 4	ABC rendező	80.-
348	C64	92/ 5	*Matek + Római számok	80.-
349	+4	92/ 6	Tesztkészítő + értékelő	80.-
350	C64	92/ 3	BASIC SYNTAXER II.	150.-
351	+4	92/ 7	Három játék	90.-
352	+4	92/ 7	Szökés	60.-
353	C64	92/ 7	Toto	120.-
354	C64	92/ 7	DIGIT analizis program	90.-
355	+4	92/ 7	SCREEN DUMP	120.-

A *-gal jelölt programok csak lemezen rendelhetők !

áborján, Fő u. 21.

obd ki a Junosztyot, és él-
ezd a színeket! Sürgősen el-
dó egy kifogástalan C–1802
pusú színes monitor kedve-
ő áron. Ugyanitt program-
sere Plus/4-re. Most 2200 db
özül választhatsz. Szekeres
éter, 1203 Budapest, Baross
. 1. II/42. Tel.: 1278-636,
279-470/180 (14 óráig).

u. 51. I/8.

Keresem A VC-1541-es lemezegység programozása c. könyvet. Tóth Gyula, 7400 Kaposvár, Füredi u. 47.

Eladó C-64-hez kártyacsalád, magnó, floppy kezelésére, FINAL III. 2700 Ft. Jakab Péter, 1046 Budapest, Török u. 25. Tel.: 16-90-779.

Amiga Sample-lemezek má-

Tagsági szám: ....

Feladandó az újság címére:
Commodore Újság
Budapest, 1388. Pf. 86.

Alulírott megrendelem a következő programokat a PÖTYÖGŐSZOLGÁLAT-tól:

PROGRAM SORSZÁMA	ÁRA
1.	,– Ft
2.	,– Ft
3.	,– Ft
4.	,– Ft
5.	,– Ft
6.	,– Ft
7.	,– Ft
8.	,– Ft
9.	,– Ft
Összesen: db	,– Ft

A programokat a SZOLGÁLAT által biztosított
lemezre (99 Ft/db)
kazettára (40 Ft/db)
az általam küldött adathordozóra kérem.
(Kérjük a megfelelő szöveg aláhúzását!)
Postaköltség (35 Ft)
A fizetés módja: személyesen — csekken — utánvéttel
(Kérjük a megfelelő szöveg aláhúzását!)
ÖSSZESEN:

A megrendelő neve: ....

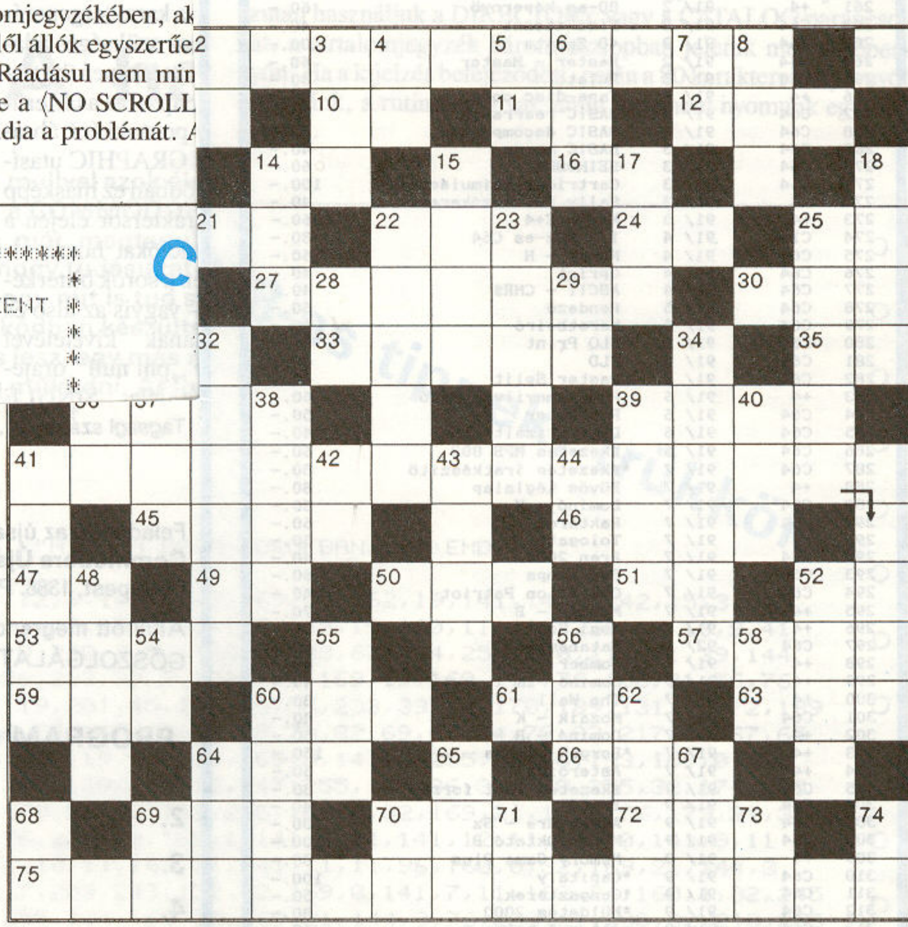
Címe: ....

GÁRDONYI GÉZA
(1863–1923)

KERESZTREJTVÉNY

Ha sok bejegyzés van egy lemez tartalomjegyzékében, ak kor a DIRECTORY parancs kiadásakor az elől állók egyszerűe kigördülnek a képernyőről, ami boszantó. Ráadásul nem min denki olyan villámkezű, hogy időben elérje a (NO SCROLL gombot. A Directory 25 Z. program megoldja a problémát. A

```
10 REM ******************************
20 REM *                            *
30 REM * DIRECTORY LAPONKENT         *
40 REM *                            *
50 REM *           C128             *
60 REM *                            *
```



vegyjele. 36. Jeruzsálem egyik dombja. 39. Együttes. 45. Ház fedele. 46. Szétszed. 47. Rénium vegyjele. 49. Részvénytársaság. 50. Imre bevézve. 51. Kettős betű. 52. Idegen kötőszó. 53. Felhők felett. 57. Visszaéget(!). 59. Veszprémi patak. 60. Földet ás. 61. Egyik megyénk. 63. Terület. 64. Bibliai alak. 66. … is marad is. 69. Mulatság. 70. Téli öltözet. 72. Ektelen akadály. 75. 1991. augusztus 10-én mutatta be a televízió 2-es csatornája (ism).

FÜGGŐLEGES: 1. Becézett férfinév. 2. E nap. 3. Felesége van. 4. Gyarapodik. 5. Tendo. 6. Lakoma. 7. Közlekedési terület. 8. Nóta. 13. Nem tisztességesen játszik. 14. Ádám felesége volt. 15. Sütemény fajta. 17. Betűvető. 18. Törőcsik művésznő utóneve. 20. Nevelős becézett férfinév. 22. Szór?. 23. … Gyula a televízió egyik riportere. 25. Állam Közép-Afrikában. 28. Sződ közepe. 29. Ilyen szomszédság is van. 32. Legújabb. 34. Nemes fémet. 37. Jelez. 38. Gondozott. 39. Királyi ülőhely. 40. Azonos a 37 függőlegessel. 41. Ékes becsület. 42. Azonos a 4 függőlegessel. 43. 1937-ben játszották ezt a magyar filmet, Fedák Sári főszereplésével 1991. augusztus 26-án mutatta be a televízió 2-es csatornája. 44. Kutya. 48. Nők réme. 52. Azonos a 48 függőlegessel. 54. Hiányos kedd. 55. Vízlelőhely. 57. Fejdísz. 58. Idegen kötőszó. 60. Párna betét. 62. Színházi alkalmazott. 64. Vízenyős hely. 65. Ilyen beszélő a telefon is. 67. Iskolai rend. 68. Előd. 69. Irányszó. 70. Kétes(!). 71. Másik irányszó. 73. Titán vegyjele.

Van-e floppyd? Akkor biztos jól jönne 1250 db lemez ingyen, tele jó játékprogramokkal C-64-re. Nem kell hozzá semmi más, csak küld el címedet és egy 15 Ft-os bélyeget. Akkor én elküldöm neked, hogy mit kell tenned. OK! Akkor a címem: CODDY, 8000 Székesfehérvár, Benke F. u. 13. fszt. 1.

HI! C-64-esek!!! Sikerprogramok lemezre, kazettára. 2 oldalnyi program csak 18 Ft, ha te küldesz lemezt, kazettára 10 Ft/db. Még valami, új képes katalógus a T.K.S.-nél. Ha igényled, akkor 20 Ft-ba kerül a képes katalógus a programok ára mellett. Címünk: T.K.S., Székesfehérvár, Benke F. u. 13. fszt. 1.

C+4, magnó, kazetták, szakkönyvek 10 000 Ft-ért eladó. 8360-8501 4000 Ft. Kiss György, 6000 Kecskemét, Bajza u. 13. IV/12.

Jó napot kazettások! Akartok programot? Csak 20 Ft-ért!! Ne hagyjátok ki! Írjatok: Rósenberg Ádám, 2100 Gödöllő, Szőlő út 28. Tel.: 10-809 vagy este 30-385.

Keresem a NEWSROOM rajzolóprogramot C-64-re lemezen. Horváth Csaba, 2400 Dunaújváros, Petőfi S. u. 48.

Eladó C-128 + 1571 + 1531 + tape + 2 joy + 12" zöld monitor, egyben vagy külön is. Gonda László, 5130 Jászapáti, Attila u. 3.

C-64 programok eladása kazettán. Olcsó árak. Felhasználói, játék- és oktatóprogramok. Salamon Péter, 4122 Gáborján, Fő u. 21.

Dobd ki a Junosztyot, és élvezd a színeket! Sürgősen eladó egy kifogástalan C–1802 típusú színes monitor kedvező áron. Ugyanitt programcsere Plus/4-re. Most 2200 db közül választhatsz. Szekeres Péter, 1203 Budapest, Baross u. 1. II/42. Tel.: 1278-636, 1279-470/180 (14 óráig).

Játékprogramokkal égetett epromokat vennék olcsón. Továbbá programokat cserélnék kazettán. Telefon: (82) 71-180.

C-64-1992-es 20 Ft, más 10 Ft, csere is, utántöltős 40—50 Ft, sex 10 Ft. Balogh Zsolt, 4031 Debrecen, István u. 51. I/8.

Keresem A VC-1541-es lemezegység programozása c. könyvet. Tóth Gyula, 7400 Kaposvár, Füredi u. 47.

Eladó C-64-hez kártyacsalád, magnó, floppy kezelésére, FINAL III. 2700 Ft. Jakab Péter, 1046 Budapest, Török u. 25. Tel.: 16-90-779.

Amiga Sample-lemezek másolása és cseréje. Válaszborítékért listát küldök. Cseréhez listát kérek. Ifj. Csörgő László, 3441 Mezőkeresztes, Ságvári u. 6. Tel.: 102.

HOBBI ELEKTRONIKA

Budapest VII. ker., Dózsa György u. 16.
(Dózsa Gy. u.—Jobbágy u. sarok)
Tel.: 1-228-892
Levélcím: 1656 Budapest, Pf.: 50.

Commodore-bővítők szaküzlete



Vidéki olvasóknak segít az egységcsomagküldő szolgáltatás: a megrendelt csomagot postán utánvéttel elküldjük.
Telefonon és levélben is rendelhet!
A HOBBI ELEKTRONIKÁHOZ nem kell hosszú levél.
Rendelését néhány sorban, egyértelműen közölje.
Levélcím: 1656. Budapest, Pf.: 50.

JÖJJÖN EL NÉZZE MEG!

Üzletünkben C-64-hez sokféle bővítő működés közben megtekinthető, megvásárolható. Minta alapján vásárolhat panelt, egységcsomagot, készre szerelt, élesztett áramkört.

ÁRAJÁNLAT:

	Egységcsomag	Készre szerelt, élesztett		Egységcsomag	Készre szerelt, élesztett
PLOFI Datassette cartridge	1200 Ft	1500 Ft	EPROM-égető	2200 Ft	6000 Ft
PLOFI Datassette/promon	1500 Ft	1800 Ft	User csatlakozó	500 Ft	—
PLOFI Fastload cartridge	1500 Ft	1900 Ft	28 lábú TEXTOOL	1180 Ft	—
PLOFI Fastload/speedtape	1800 Ft	2200 Ft	Égető szoftver lemezen	1300 Ft	—
PLOFI Simon's cartridge	1600 Ft	2000 Ft	Égető szoftvercartridge	2200 Ft	3000 Ft
FINAL III cartridge	—	3450 Ft	EPROM-bank	4000 Ft	5000 Ft
ACTION Replay V., VI., VII.	—	3450 Ft	C-64 IC teszter	3500 Ft	6000 Ft
Fényceruza szoftverrel	950 Ft	—	Datassette—datassette gyorsmásoló	600 Ft	—
Hangdigitalizáló	750 Ft	—	Datassette fejbeállító	350 Ft	—
Hangkapcsoló + szoftver	300 Ft	—	CPU-stop + reset	400 Ft	—

Az árak tartalmazzák a 25 % ÁFA-t!

Az Országos Commodore Egyesület szolgáltatásai

Egyesületi tagoknak 20% kedvezmény:

VC—20 memóriabővítés 3—27 kByte-os:	kiépítéstől függő
C—16, C—116 memóriájának bővítése 64 kByte-ra:	3500 Ft
C—16 belső 16 kByte-os EPROM bővítés:	1450 Ft
C—16 belső 32 kByte-os EPROM bővítés:	2900 Ft
C—16 belső 8 kByte-os SOFT—ROM bővítés:	2800 Ft
C—16 belső 32 kByte-os SOFT—ROM bővítés:	4000 Ft
C—16 8 kByte-ról 32 kByte-ra átalakítás:	2000 Ft
C—16 és 1541 kompatibilis lemezegység párhuzamosítása:	3200 Ft
SOFTROM modul 32K, kikapcsoláskor sem felejt C-16, C-116, +4	5000 Ft
FÉK C—16, C—116, +4 potméteres sebességváltoztatás 0%-tól 100%-ig fokozatmentesen	2000 Ft
TTL IC-teszter (Cartridge+lemezen a program)	4300 Ft
+4, C—16, C—116 UNI—ROM modul különféle kiépítésekben:	
— 8 kByte SOFT—ROM	3400 Ft
— 16 kByte SOFT—ROM	4000 Ft
— 8 kByte SOFT—ROM 16 kByte EPROM	4400 Ft
— 16 kByte SOFT—ROM 16 kByte EPROM	5000 Ft
— 16 kByte EPROM	2200 Ft

Egyesületi tagoknak 10% kedvezmény:

C64-be átkapcsolható új operációs rendszer (Speed) + reset beépítése:	2000 Ft
1541 kompatibilis lemezegységbe Speeddos beépítése (átkapcsolhatóan) 40 TRACK (+ 85 blokk/lemezoldal), valamint párhuzamos 15 pólusú Canon csatlakozó beépítése:	2000 Ft
C64 USER-port 1541-es lemezegység összekötő párhuzamos kábel:	1300 Ft
1541 kompatibilis lemezegységbe elektronikus lemezlyukasztó beépítése:	800 Ft
PAGEFOX magyar ékezetes szövegszerkesztővel rendelkező cartridge: (Epson típusú nyomtató min. 640 képpontos szükséges a nyomtatáshoz)	6000 Ft
FASTLOAD (lemezes gyorstöltő, másoló, monitor)	1400 Ft
TTL IC-TESZTER cartridge + program	4300 Ft
288/256 Kbyte-os eprombank (vezérlő eprommal)	5340 Ft
Eprométegő (2716-tól 27256-ig)	4300 Ft
8—16 Kbyte-os epromkártya (cartridge, eprom nélkül)	600 Ft
C64-hez tároló oszcilloszkóp (párhuzamos kábel nélkül)	7500 Ft
A háttértárakhoz epromok programozása (kész programok, vagy saját, hozott programok beégetésével) egységesen:	500 Ft
C64 bővítő-port elosztó (egyszerre 4 db cartridge lehet a gépben, melyeket gombnyomásra lehet kapcsolni)	7500 Ft
C64 USER — CENTRONICS nyomtatókábel (GEOS kábel)	1500 Ft
256K RAM-diszk	13000 Ft
64/256K RAM-diszk	9000 Ft
256K RAM-diszk (RAM-ok nélkül)	7500 Ft
64 Kbyte-os cartridge komplett programokkal, vagy igény szerint összeállítva	2500 Ft

A fenti bővítések megrendelhetők levélben, vagy az OCE irodájában személyesen.
Ha személyesen kívánja megrendelni, kérjük, előtte telefonáljon.

Árainkat az alkatrészárak változásai befolyásolhatják.

input: **MAX** output: **maximum**

A TUNGSRAM-MAX mágneslemez japán és amerikai alapanyagból, amerikai technológiával, high-tech berendezéseken készül. Minden egyes mágneslemez hibamentességét a teljes felület számítógépes mérőrendszerrel történő tesztelése garantálja.

TUNGSRAM-MAX mágneslemezek

Standard	Formázott	Színes lemezek műanyag dobozban
5,25" TM 2S2D 53 Ft	5,25" TMF 2S2D 61 Ft	5,25" TMP 2S2D 63 Ft
5,25" TM 2SHD 77 Ft	5,25" TMF 2SHD 87 Ft	5,25" TMP 2SHD 88 Ft
3,5" TM 2S2D 88 Ft	3,5" TMF 2S2D 102 Ft	3,5" TMP 2S2D 99 Ft
3,5" TM 2SHD 155 Ft	3,5" TMF 2SHD 173 Ft	3,5" TMP 2SHD 167 Ft
Árainkhoz ÁFA-t számítunk!		

- ☐ CSEREGARANCIA: 20 ÉV VAGY 20 MILLIÓ FORDULAT!
- ☐ Mágneslemezek: no name és bulk csomagolásban is!
- ☐ Tárolódobozok, tisztítókészletek 3,5" és 5,25" méretben.
- ☐ Szoftvermásolás profi gépeken, írásvédő kivágás nélkül is!
- ☐ Viszonteladóknak 20% engedmény!
- ☐ Szoftverkészítőknek, nagyfelhasználóknak, diákoknak rendkívüli kedvezmények!
- ☐ Színes és formattált mágneslemezek, tárolódobozok, festékszalagok árusítása, szoftvermásolás és csomagolás, címkézés a szoftverkészítő igénye szerint.
- ☐ Kérje részletes árlistánkat!
- ☐ Tungsram Magnetic Media Budapest IV., Váci út 77. Tel.: 160-2233 Fax: 160-0925



TUNGSRAM-MAX®

KÉPVISELETEINK:

Agromark Kft. — Hódmezővásárhely, Andrássy út 50. Tel.: 06-62-41695

Comtrade Kft. — Pécs, Majorossy u. 36. Tel.:06-72-26063

M és K Bt. — Kecskemét, Bajcsy-Zsilinszky u. 5. Tel.: 06-76-21878

PGM Computer Kft. — Szeged, Csongrádi sugárút 22. Tel.: 06-62-14380

Számker Kft. — Zalaegerszeg, Rákóczi út 4—8. Tel: 06-92-14500/144

Transzfer Kft. — Nyíregyháza, Hősök tere 7. Tel.: 06-42-10481

HIVATALOS DEALEREINK:

Novotrade 2C Kft. — Budapest XIII., Balzac u. 35. Tel.: 140-2954

Radiant Kft. — Budapest XIV., Francia út 11. Tel.: 252-1999/266

Westing Iroda — 1149 Budapest, Róna köz 12. Tel.: 163-7916

TUNGSRAM MAGNETIC MEDIA RT.
H-1340 Budapest, Váci út 77.
Tel.: 160-2233, Fax: 160-0925

TUNGSRAM MAGNETIC MEDIA

SZERENCSEJÁTÉK RT

A magyar olimpiai csapat arany fokozatú támogatója

Ára: 68 Ft

ACOMP

ACOMP Számítástechnikai Kft.

1141 Budapest, Álmos vezér útja 17., Tel.: 183-1817, Fax: 251-2523

A j á n l a t a i n k

Commodore Amiga 500	35 920 Ft
Commodore Amiga 500 Plus	43 992 Ft
Commodore Amiga 2000	69 999 Ft
Commodore 1084s Stereo-Color monitor	26 320 Ft
Commodore A-520 TV-Modulator	2 800 Ft
Commodore C-64 II	11 680 Ft
Commodore 1541 II Floppy	13 280 Ft
Commodore 1802 monitor	19 920 Ft
Commodore 1352 mouse (eredeti Amiga)	4 792 Ft
Philips 8833 II. Stereo-Color monitor	26 320 Ft
Telmex 512 Kb órás memóriabővítő	4 000 Ft
2.0 Mb órás memóriabővítő	11 920 Ft
4.0 Mb órás memóriabővítő	19 920 Ft
1 Mb bővítő Amiga 500 Plus-hoz	6 320 Ft
Alfadata 3.5" külső floppy drive	7 920 Ft
Roctec 5.25" külső floppy drive	11 920 Ft
Epson LQ-200 24 tűs printer	31 920 Ft
Noris DR 1535 Datassette	2 000 Ft
Quickshot II joystick	552 Ft
Quickshot II plus joystick	712 Ft
Quickshot II turbo joystick	552 Ft
Quickshot QS-113 analóg joystick	792 Ft
Quickshot QS-123 analóg joystick	1 032 Ft
NoName 3.5" DSDD lemez	400 Ft
NoName 3.5" DSHD lemez	640 Ft
NoName 5.25" DSDD lemez	216 Ft
NoName 5.25" DSHD lemez	400 Ft
TDK MF-2DD 3.5" lemez	792 Ft
TDK MF-2DD 3.5" lemez 5 darabos	400 Ft
Maxell 3.5" MF2-DD lemez	792 Ft
Maxell 3.5" MF2-HD lemez	1 432 Ft

Maxell 5.25" MD2-D lemez	472 Ft
Maxell 5.25" MD2-HD lemez	792 Ft
Profex 3.5" DSDD lemez	544 Ft
Profex 5.25" DSDD lemez	312 Ft
Profex 5.25" DSHD lemez	544 Ft
Fuji 3.5" MF 2D lemez	792 Ft
Fuji 5.25" MD 2DD lemez	472 Ft
Fuji 5.25" MD 2HD lemez	792 Ft
Noris C15 C-64 kazetta 5 db	312 Ft
Action Replay MK III	12 792 Ft
4 Player Adapter (4 Joystick csatlakozó)	1 500 Ft
PAN C1531 Mouse	2 232 Ft
PAN C64/C128 Mouse	2 232 Ft
PAN Amiga/Atari Mouse	2 232 Ft
Noris Maus M3 Amiga	2 000 Ft
Noris Maus M1 C-64	1 992 Ft
Noris MB 10 3.5" lemeztartó	136 Ft
Noris MB 40 3.5" lemeztartó	552 Ft
Noris MB 80 3.5" lemeztartó	712 Ft
Noris DB 10 5.25" lemeztartó	136 Ft
Noris DB 50 5.25" lemeztartó	552 Ft
Noris DB 100 5.25" lemeztartó	712 Ft
Noris Amiga 500 porvédő	792 Ft
Noris C-64 II porvédő	632 Ft
Noris MF 14 C 14" monitorfilter	1 112 Ft
Maus Pad	392 Ft
Midi Amiga Interface	2 072 Ft
Handyscanner Amigához	10 392 Ft
Boot Selector Amigához	1 032 Ft
Thunder board hangkártya	15 120 Ft

Viszonteladóknak óriási árkedvezmény! Áraink az ÁFA-t nem tartalmazzák!

Nyitvatartás 9–18 óráig, szombaton: 9–12 óráig.

Vidéki vásárlóknak utánvételes csomagküldő szolgálat!

ACOMP Számítástechnikai Kft.

1141 Budapest, Álmos vezér útja 17., Tel.: 183-1817, Fax: 251-2523